パソコンプログラマのためのクリエイティブマガジン

月刊プログラマーズ・ページ

平成3年4月2日第三種郵便物認可　1992年4月1日発行（毎月1回18日）第2巻第4号通巻16号

Programmer's Page

MONTHLY MAGAZINE FOR PROGRAMMERS

1992 4 APRIL

定価980円

特集
新しいDOS環境を探検する
MS-DOS 5.0, DOS/Vプログラミングの知識と技術

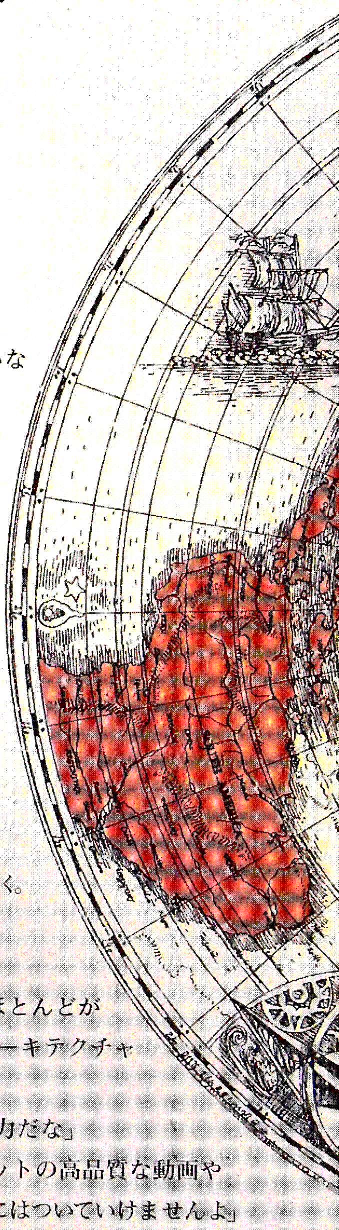
付録:5"2HDフロッピーディスク

本誌掲載プログラム／綴じ込み付録テキストファイル（Microsoft Macro Assembler Professional Development System Version 6.0マクロ／ディレクティブ・演算子一覧）／クイズ・ランダムスクリーン解読ツール／テキストファイル高速ページャー「MIEL Ver.2.41」

SE
Shoeisha Co., Ltd.

世界標準物語　第二話

14本目の煙草



とある企業のオフィス。二人の男がテーブルをはさみ、真剣な顔で向き合っている。
テーブルの上の灰皿には、すでに12本の煙草の吸殻がある。

A部長「それじゃあキミはどうしても、AXがいいというんだね？」

X主任「はい、どうしてもAXがいいんです。今回だけは一歩も譲れません」

短い沈黙。A部長、13本目の煙草を取り出し、おもむろに火をつける。

A部長「しかし、うちの社内でAXを使っているのは、まだキミを含めて数人しかいないんだ。他のものが使いこなせるかね」

X主任「使いこなせるもなにも、操作自体は全く変わらないんです。違うのは、パソコン内部のアーキテクチャです。AXは、事実上の世界標準機であるIBMPC/ATと同じオープンアーキテクチャを採用しているので、世界に6万本以上といわれるIBMPC用ソフトがそのまま使えるんです。今、うちの社内で使っているパソコンは、日本では一応、標準機ということになってますが、閉ざされた、メーカー独自のアーキテクチャを採用しているので、そういうことはないんですよ」

A部長「IBMPCは、そんなに世界中で使われているのか？」

X主任、待ってましたとばかりに、一枚の世界地図をとりだす。

X主任「見てください。アメリカはもちろん、カナダ、メキシコ、ブラジル、イギリス、フランス、ドイツ、ソ連、中国…………。IBMPCとその互換機は日本以外のほとんどの国でトップシェアを占めているのが現実です」

X主任、つぎつぎに国名をあげながら、猛スピードでその国の位置を赤鉛筆で塗りつぶしていく。

A部長、思わず大きな溜息とともに煙を吐く。

A部長「うーん、これはすごい。日本はまさに世界の異端児ってわけだ」

X主任「MS-Windows、LAN、DTPソフトなど、現在我々が使っているソフトのほとんどが海外でIBMPC用として開発されたものです。日本語への移植も、AXなら基本アーキテクチャが同じなので即座に行えるというわけです」

A部長「GUI環境を実現するWindows3.0対応ソフトがすぐ使えるのは、確かに魅力だな」

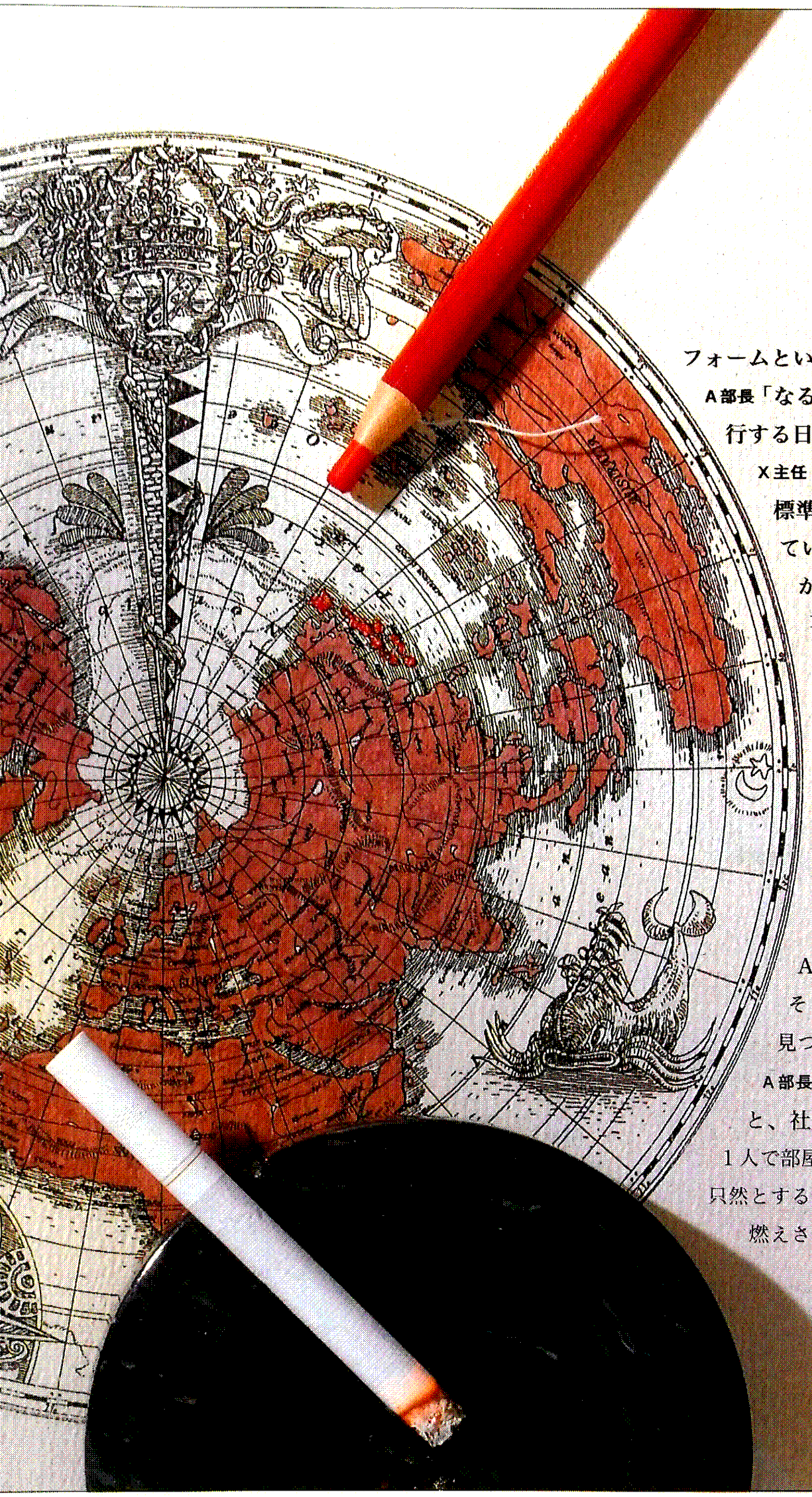
X主任「それにAXはVGAに対応していますからね。これらの画像は、640X480ドットの高品質な動画やアウトラインフォントを扱えるVGAモードでなくちゃ、世界のマルチメディアの流れにはついていけませんよ」

A部長、煙草を揉み消し、身を乗り出す。

X主任「オープンアーキテクチャのAXこそが、ネットワークコンピューティングを可能にする共通プラット

世界の標準機を、日本の標準機に。
我々が、パソコンの新しい時代を拓きます。

ACER　Canon
MITSUBISHI　NCR　OKI　olivetti　SANYO



THE ARCHITECTURE EXTENDED-AX

フォームといえるんです」

A部長「なるほど、日本市場が海外市場と同時進行する日も近いわけだ」

X主任「部長、事実上の世界標準機が、日本標準機になる日が、もうすぐそこまで来ているんです。新しい時代に備えて、今が決断の時です」

再び沈黙。A部長、14本目の煙草に火をつける。

その時ドアをノックする音がして、部長の秘書が現われる。

秘書「部長、会議のお時間です。もう社長がいらしてますので急いでください」

A部長「しまった、もうそんな時間か。すぐに行くと伝えてくれ」

A部長、慌てて立ち上がりドアに向かう。そこで振返って、X主任の目をまっすぐ見つめて言う。

A部長「よしわかった。……AXしかないと、社長に言うことにしよう」

1人で部屋に残されたX主任は信じられぬ思いで只然とする。が、すぐに満面に笑みを浮かべると、燃えさしの14本目の煙草を灰皿に押しつけた。

※MS-Windows 3.0は米国Microsoft社の登録商標です。
※IBM PC、IBM PC/AT、VGAは、米国IBM社の商標です。

CASIO　HEWLETT PACKARD　HITACHI　KYOCERA
SHARP　SONY　UNISYS

社名アルファベット順

お問合わせ：株式会社 翔泳社
TEL. 03-3263-0447

Brief

Full Screen Editor 日本語版

全米のプログラマが認めた実力！
アメリカでは「ブリーフ」が
プログラミング用エディターとして
最も多く利用されています。

C言語で組めるマクロ！

Version UP!

V.3.0 いよいよ登場。
プログラミングエディター新世紀。

よりパワフルで、よりエレガントな プログラミングのために……。

●C言語で組めるマクロ 新機能

使い慣れているC言語でマクロが組めますから、習得が簡単でカスタマイズも思いのままです。

●マクロのスクリーン・デバッガ 新機能

ソースコードでデバックできます。もちろんブレークポイントやウォッチ・ドッグもセットできます。

●テンプレート機能〈構文の自動生成〉 新機能

C、Ada、Basic、COBOL、FORTRAN、MODULA2、Pascalで構文を自動生成します。

●300回ものアンドゥ

各ファイルごとに300までの操作を元にもどすことができますので、やり直しの面倒を気にせず、安心して編集できます。

●安全編集

Briefは、一定時間キーを操作しないこと最新の編集内容をファイルに自動保存します。だから編集中に電源が切れても安心です。

●その他の主な新機能

キー操作のファイルへのセーブとリプレイ、処理可能行数を拡張、リジューム機能、Ctrlキー・アサイン（オプション）、メニュー（オプション）etc。

アメリカでも日本でも高い評価

●プログラミング向きエディターを探している人は、もう探す必要はない。（Jerry Pournelle BYTE）
●ソフト開発には強力なツール、300回のアンドゥは大きな特徴。（日経バイト）
●既存のエディターに不満を覚えたら、これしかない。（Oh! FM）
●まるで夢のようなアンドゥだ――おもわず間違うのを期待してしまいそうだ。（The C Users Journal）

■主な機能

◆55ウィンドウ◆無限のファイル◆自動コンパイル◆自動エラー・ジャンプ◆自動インデント◆自動テンプレート◆バッファ書出 切換 削除 リスト表示◆ファイル名の指定 表示 変更◆バックアップ保存◆特殊文字入力◆2行を1行に連結◆ファイル単位の挿入◆増分検索◆各種マーク 始・終点の入換◆ブロックのカット コピー ペースト 印刷 保存 左右へのシフト◆対応する括弧へ移動◆C言語のコメント化 非コメント化◆DOSコマンドの実行◆キー操作記憶 実行◆入力ヒストリー◆英文テキスト整形◆タブのスペース挿入◆拡張子毎に異なった設定ができるタブ◆検索置換パターンの画面からの入力◆英字の大小文字変換 その他

定価 48,000円
（消費税は含まず）

■5機種対応パッケージ

（東芝、NEC、AX協議会、IBM、富士通）

●5 2DD、3.5 2DDデュアルメディア●MS-DOS3.1以降に対応●英語DOSでは使用できませんが、英語版BRIEFもセット販売いたしております。●お急ぎの方、上記以外のメディア希望の方、英語BRIEFのセット販売希望の方はユーザー・サポート係までご連絡下さい。

日本システム株式会社

●お問合せは　東京都立川市曙町1-18-2一清ビル立川事業所Briefユーザー・サポート係
TEL0425-23-4469　FAX0425-27-6127

〈資料請求番号001〉

PLUS ONE
PERSONAL ENVIRONMENT & TECHNOLOGY

AT互換機をDOS/Vマシンにシステムアップ

日本語DOS/Vキーボード

型番　PKB-106J……………………………標準小売価格 19,800円

特　徴

●日本IBM社の002型キーボード準拠のJIS配列キーを採用し、DOS/V対応のあらゆるアプリケーションに対応しています。●キートップのカナ刻印はカタカナを採用しています。●コンパクトなキーボードサイズを実現しています。(幅424mm×奥行152mm×高さ39mm)　●CPUユニットとの接続端子は、ATキーボード端子(DIN5ピン)ですが、AT端子からPS/2端子への端子変換コネクタも付属しており、PS2端子のマシンでも使用可能です。(なお、キーボードにスピーカーは付いておりません)

DOS/V・日本語Windows 3.0対応SVGAボード

型番　PVC-1000……………………………………………………………………標準小売価格 48,000円

VRAMを1MB搭載し、数多くのグラフィックモードを持ち、高度な表現力を実現しています。ハード的な最大解像度は1024×768ドット状態で262,144色中256色同時発色が可能で、DOS/Vマシンのグラフィックボードに最適です。 SUPER VGAは従来のVGA・EGAと完全上位互換です。CGA・MDA・Herculesに対してもBIOSレベルで互換性があります。●16ビット(AT)バス対応、マルチメディア等の対応ボード接続端子フューチャコネクタ装備。 ●ボードサイズは、ハーフサイズ(奥行181mm×高さ104mm)で出力端子は、シュリンク15ピン(IBM標準VGA端子互換)

添付ユーティリティソフト

●日本語WINDOWS用に高解像度ディスプレイドライバーを添付。1024×768ドット(256色/16色)、800×600ドット(256色/16色)、640×480ドット(256色)　●英文アプリケーション用には主要アプリケーションに対し、SVGA対応ユーティリティ添付。●強化高速グラフィックBIOS添付(TSR型ソフトで、386SX CPU以上のみ対応)

超高速SCSI II インターフェース

型番　PBT-542B……………………………………………………………………標準小売価格 94,800円

●高速なSCSI II 対応のHDDドライブに対応。●バースト転送時最大5MB/Secの転送レートを実現(実際の転送レートはドライブの性能に依存)●HDDドライブとCPUユニットの両方のデータ転送効率を最大に活用するために、80186CPUを搭載。●内蔵ドライブ2基、外部ドライブ5基まで対応。ただし、通常BIOSでは、HDDは2基の対応となっています。それ以上のドライブを付ける場合には、サポートユーティリティが必要となります。●ネットワークホスト対応で、デュープレキシング(SCSIボード2枚使用)、ミラーリングが可能です。●内蔵フロッピーコントローラ機能搭載、 ●ボードサイズは、(奥行193mm×高さ104mm)、外部端子はDサブ50ピン。

内蔵型大容量・高速SCSI IIハードディスク

容量340MBハードディスク/型番　PHD-340SC……………………………標準小売価格 340,000円
容量530MBハードディスク/型番　PHD-530SC……………………………標準小売価格 450,000円

大容量タイプのハードディスクで、AT互換機の5"または3.5"のハードディスク格納用スペースに格納します。シークタイムは12msで、高速SCSI II ボードPBT-542Bと組合せた場合には、データ転送レートは約1.2MB/sとなります。

バスマウスセット

型番　PBM-200B……………………………………………………………………標準小売価格 9,800円

マイクロソフト・バスマウス互換のバスマウスセットです。シリアルマウス等にくらべマウスの反応速度が高く、Windows等のGUI環境でのレスポンスが向上します。

セット内容

- ●XT(8ビット)/AT(16ビット)バスマウス用インターフェースボード
- ●ミニDIN9ピン仕様のバスマウス(250 DPI)
- ●グラフィックエディタ等のユーティリティ付属(メディアは5"フロッピーで提供)

●御購入の際、消費税が付加されますので御了承ください。
●社名・製品名はすべて各社の商標または登録商標です。　●ご購入の際は、購入年月日・販売店名など所定の事項を記入した保証書を必ず御受け取り下さい。

PLUS ONE
関西電機株式会社プラスワン
〒101 東京都千代田区岩本町3－2－10 SN岩本町ビル
プラスワン問い合せ TEL.03-5687-6912
テクニカルサポート TEL.03-5687-6913
プラスワン つくば TEL.0298-56-8851

〈資料請求番号002〉

巻末特別企画
Tiny C Journal

米国「The C Users Journal」「Windows™/DOS DEVELOPER'S JOURNAL」提携誌
月刊 Cジャーナル
C言語を主体とするコンピュータプログラマのための月刊情報誌

C JOURNAL

C JOURNALは、毎月8日全国の書店でお求めになれます　定価980円（本体951円）

4
APR.1992

3月8日発売

特集
ポータビリティ

■移植とポータビリティ
■CライブラリをC++で使う
■ポータブルなVMS方式の行入カルーチン
■OS/2用MIDIデバイス・ドライバ

第2特集　UNIXへの招待

特別記事
▶ソフトウエアの特許化に反対する
▶DOSエクステンダー32ビットのメモリとパワー
▶Bounds-Checker

連載
●STANDARD C
●Dr. C's Pointers
●Questions & Answers
●MINIX通信
●GUI環境
●プログラマの妻たち
●BART EXPRESS
●GNU MS-DOSへの移植
●たのむぜ！DOS/V！
●MacでSmalltalk［最終回］
新連載 TECH Tips

付録 DISK
GNU plot
数式データのプロットプログラムソース付最新版
GNU gcc　386エクステンダ移植バージョン
GNUダイジェスト　1992年1月発行第一版

株式会社ビレッジセンター
出版局　〒101 東京都千代田区三崎町3－1－16 神田アメレックスビル2F
●営業部：03-3221-1825●編集部：03-3239-1950

98シリーズで使う
Visual Basic入門
塚本吉彦著
株式会社ビレッジセンター出版局

98シリーズで使う　好評発売中
Visual Basic入門
塚本吉彦著　A5判208頁　定価980円

Visual Basic（英語版）は、1991年6月にマイクロソフト社から発売されました。1990年末から1991年前半にかけてWindowsのプログラミングツールが続々と発売になりました。なかでも、マイクロソフト社が満を持して送り出したこのVisual Basicは、すぐれたツールであると思います。Windowsプログラミングの難点であったユーザインターフェイスに関する煩わしい作業をWindowsとVisual Basicが引き受けてくれるのです。その操作性の簡便さは、プログラマだけでなく、一般のユーザにも使えるWindowsのプログラミング環境を提供しています。また39,800円という価格は、十分にエンドユーザの手の届く範囲にあり、その好調な売れ行きに拍車をかけています。本書は、このVisual Basicがどのようなソフトであるかを紹介し、現在販売されている英語版をPC-9800シリーズで使うことを前提に、書いてあります。

パソコンの聖地バークレイと14人のハッカーたちの半生　■近刊案内

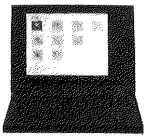
Berkeley Hackers
バークレイ・ハッカーズ
Zachary Margulis & Carol Palecki 著
広谷 渉訳　A5判250頁　予価1500円

発売：株式会社ビレッジセンター 出版局
〒101 東京都千代田区三崎町3－1－16 神田アメレックスビル2F
●営業部：03-3221-1825●編集部：03-3239-1950●FAX：03-3221-1768

IBM Authorized Dealer

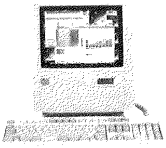
Net Ware取扱開始！ NEWマシンぞくぞく大量入荷!!

PS/55 note



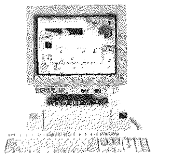
大容量80MBハードディスク内蔵 5lll SV6 5lll SA8
パワフルなDOS/Vノート
80386SX(16MHz)2MB FDD内蔵

PS 55Z 30U



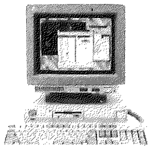
XGA採用パーソナルディスクトップ
80386SX(20MHz)2MB or 6MB
パソコンらしくないパソコン?
オシャレ

PS 55Z 10S



386SX搭載の
スタンダードPC
DOSV.VGA.ATバス採用
80386SX(20MHz)2MB or 6MB

PS 55Z 10T



PS/55Z
最強パフォーマンスPC
XGA高解像度・大容量
80386(20MHz)2MB or 6MB

ディスプレイディスパッチドライバー by C·F·Computing

今話題のS3ボードやTiGAボードに付属の英語ドライバーで日本語が使える!! 特価¥3,800

VGA DOS/Vの必須アイテムは早いVGAアダプタです。

ORCHID FAHRENHEIT 1280 1MB ¥89,800
ATI GRAPHICS VANTAGE 1MB ¥89,800
ATI GRAPHICS ULTRA 1MB ¥158,000
STB WIND/X(マウス付) 1MB ¥82,800
STB 1280 1MB ¥108,000
STB 1280 2MB ¥138,000
ET-4000 1MB ¥29,800

CREATIVELABS
マルチメディア対応のSOUND BLASTERは必需品です！ 特価提供中
SOUND BLASTER

これ1つでマルチメディアが楽しめます。
内蔵CD-ROM
SOUND BLASTER PRO
MIDI KIT
マルチメディアイクステンション他ソフト付
UPGRADE KIT ~~¥184,800~~

ステレオ22FM音源
CD-ROMインターフェイス他
SOUND BLASTER互換
Multimedia Sound Standard
PRO ~~¥57,800~~

11FM音源
MIDIジョイスティックパワーアンプマイク
入力 各種ソフト付 V2.0 ~~¥34,800~~

PS/55等のマイクロチャンネル版です。
MCA ~~¥64,800~~

VGAモニター

DOS/V対応やWINDOWS用として
高解像大画面のモニターがほしいあなた、
何を選びますか?

NANAO T660i (ケーブル別)
20"0.31㎜ 30〜78kHz 1280×1024 ~~¥598,000~~

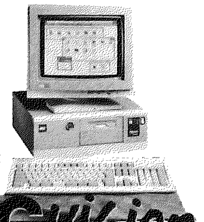
三菱 XC-98V3
14"0.28㎜ 98/VGA 640×480 ¥78,000

TATUNG(マイコン搭載)
17" 0.28㎜ 29〜65kHz 1280×1024
超特価 ¥180,000

AST ASTから DOS/Vマシン

RESEARCH JAPAN K.K.

新登場



80386SX/20MHz
FDDモデル ~~¥308,000~~
40MBモデル ~~¥355,000~~

- 高速32ビットCPUを採用。
- メインメモリ2MBを標準搭載
- 80387数値演算プロセッサの搭載が可能。
- IBM PC ATスロットを5基装備。

PCvision

マルチメディア対応高速CD-ROMドライブ

- 標準SCSI(最大7台まで同時接続可能)
- 高速アクセス(340ms)、64KBキャッシュ内蔵
- CD-ROM XA対応
- 市販オーディオディスク演奏可能

	ドライブのみ	AT I/F付	PS/2 I/F付
内蔵モデル	¥125,000	¥145,000	¥158,000
外付モデル	¥135,000	¥155,000	¥168,000

特価提供中

DOS/Vノートおすすめセット 限定

- CPU：80386SX/20MHz
- RAM：3MB
- FDD：3.5"×1
- HDD：40MB
- 外部VGA出力
- A4ファイルサイズ

＋ DOS J5.0/V ＋ シリアルマウス ＋ 一太郎dash

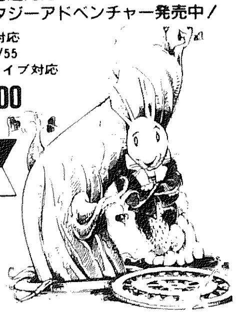
何と！何と！の 超特価¥278,000

the Manhole (日本語版)

それはマンホールからはじまった……
ゲームを超えた
ファンタジーアドベンチャー発売中！

DOS/V対応
IBM PS/55
テラドライブ対応

¥9,800



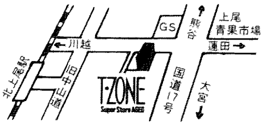
OPEN

スーパー型パソコンストアー
上尾にオープン!

新感覚の コンピュータショップ
オープニングセール
'92年2月29日〜4月12日

1F Apple. IBM. NEC用ソフトウエア
2F Apple. IBM. NEC用ハードウエア&アクセサリー
☆今までのパソコンショップとはちがう在庫量、価格。

T·ZONE Super Store AGEO



営業時間 AM10:30〜PM7:00
〒362 埼玉県上尾市緑丘3-5-28
TEL·048-774-0511 FAX·048-774-0823

Mitac 3060D

106キーセット
超特価 ¥258,000

101キーセット+DOS5.0
超特価 ¥258,000

共通仕様
80386SX 16MHz、3.5" 1.44MB×1、HD 40MB、スーパーVGA(512KB)、スーパーVGAモニター

DOS/V版 パソコン電子手帳くん 新登場

¥14,800

シャープの電子手帳とDOS/Vマシンの間で
データのやりとりができるようになりました。
対応している電子手帳
PA6000、6500、7000、7500、8500、8600シリーズ。

80486/33MHz made in USA 超特価¥498,000

80486/33MHz
RAM 8MB(max 32MB)
128KBキャッシュ(max 256KB)
3.5"1.44MB×1/5"1.2MB×1
200MB IDE HDD
スーパーVGA ET-4000 W/1MB

2シリアル、1パラレル、1ゲーム
拡張スロット×8
ドライブベイ×6
ミニタワー、2ボタンマウス
106キーボード

80386/33MHz ノート made in TAIWAN

限定20台 超特価¥398,000 最強ノート

80386/33MHz
RAM 2MB(max 16MB)
32KBキャッシュ
3.5"1.44MB×1
40MB HDD
32階調VGAバックライト液晶

外部VGA出力
1シリアル、1パラレル
外部テンキー付属
外部101キーボード、外部FDDポート
ニッカドバッテリー(充電2時間、2時間使用)
280W×220D×54H 4kg

80386/33MHz セット 超特価¥398,000

RAM 4MB
64KBキャッシュ
3.5"×1、5"×1
80MB HDD
スーパーVGA 1MB SVGAモニター

2シリアル、1パラレル
1ゲーム
マウス
101キー

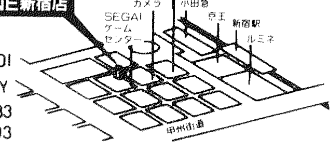
Microsoft BALLPOINT MOUSE 超特価¥28,000

VGA信号をビデオに変換するコンバーター 超特価¥98,000
320×200〜640×480の信号をS-VHS、ビデオに変換して、テレビやビデオに接続できます。

MINI-SCSI 超特価¥29,800
プリンタポートにつなぐだけでSCSIデバイスが使えるようになります。

87MB HDDセット 超特価¥98,000

T·ZONE新宿店 ☎03 3342-1901



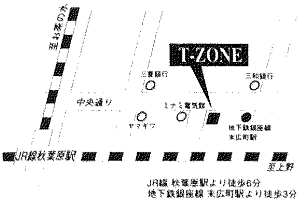
1F APPLE&NEC
03-3342-1901
2F IBM PS/FACTORY
03-3342-5533
FAX 03-3342-1903
〒160 東京都新宿区西新宿1-12-18

T·ZONE PS PLAZA

IBMゾーン
3F ハードウエア
4F ソフトウエア

営業時間
AM10:30〜PM7:00
株式会社 亜土電子工業

☎03-3257-2652
FAX 03-3257-2661
〒101 東京都千代田区外神田4-4-1



T·ZONEスタッフ募集 IBM、Apple、無線のスタッフを募集中です。アルバイトも募集中！ ☎03-3257-2690

☆T·ZONEの7Fは専門書籍(国内・外)・データーブックの専門フロアです。大量在庫中！ ほしい本は何でも揃います。

〈資料請求番号003〉

Programmer's Page
MONTHLY MAGAZINE FOR PROGRAMMERS

特集

新しいDOS環境を探検する

MS-DOS 5.0, DOS/V プログラミングの知識と技術

基本に忠実なきれいなプログラミングを!
MS-DOS5.0の新機能とプログラミング　マイクロソフト(株)中嶋雄一　2

(株)アドミラルシステム　堀　正明氏に聞く
アプリケーションプログラムをMS-DOS5.0に対応させる　9

インストールからタスクの切り換えまで
NEC MS-DOS5.0の新機能の概要　志村拓　16

MS-DOS5.0プログラミングのために――プログラマーズリファレンスを概観する　28

DOS/V プログラミングの基礎を探る
DOS/Vとはなにか？　柳澤健二　30

プログラマの理想郷を探る
DOS/Vパソコンの開発環境　田中亘　42

高級言語に近づくマクロ機能
MASM6.0の機能と特徴　秋山登　54

IC 千夜一夜　第10回
ミニコンピュータのマイクロコンピュータ化　虎岩 登　74

プログラム「遊」工房　拾伍乃巻
trackrw/フロッピーディスクの直接読み書き　谷口典生　78

イカす BASIC プログラミング入門　第4回
イカすアレンジ編　酒井法雄　85

実践C塾　アルゴリズム篇　第4回
サーチ・その2　加野島英渡・山本博昭　90

98活用アセンブラ技法　ブロックデバイス編　第1回
不揮発性 RAMDISK ドライバの作成　星野操　100

付録

5"2HD フロッピーディスク

綴じ込み付録

Microsoft Macro Assembler Professional Development System Version 6.0　マクロ／ディレクティブ／演算子一覧

表紙イラスト：日暮修一
表紙 AD：佐々木幹夫(SE)

コンピュータのなかのSF　第1回
コンピュータ創世記　志賀隆生　70
プログラマの道具学　第16回
胃腸薬　祐安重夫　72
PRODUCTS REVIEW
Microsoft FORTRAN Professional Development System Version 5.1　海老原浩之　113
Book Review「笑うコンピューター」　中村正三郎　118
本のアンテナ　古田島義和
佐々木哲のU.S.プロダクツウォッチ　120
西川和久の希望的観測　122
コンピュータの法律　熊谷信太郎　124
アキバ短信
プログラマの仕事場　第13回　株式会社エーエスピー　126

INFORMATION PAGE　129
NEWS & NEW PRODUCTS　130
Programmer's Program PARTNER　134
オンラインソフト採集帳16「鮪ペイント」　136
ドラゴン・ポスト　138
付録ディスクのご案内　140
バーチャルスペースガイド　141
読者プレゼント
クイズ・ランダムスクリーン　142
SE-NETからのお知らせ
バックナンバーのご案内　143

広告索引　資料請求番号

AXメーカー連合広告
日本システム（株）……001
[illegible]（株）……[illegible]
ビジョンセンター
T-ZONE……00[illegible]
丸文（株）
ボーランド（株）……P04

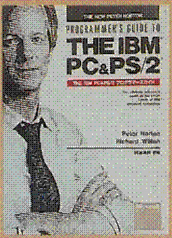
人材募集広告

（株）[illegible]
マイクロソ[illegible]（株）
（株）[illegible]ビジョン
（掲載順）

1992 APRIL No.16

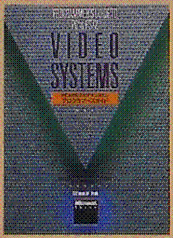
翔泳社が贈るプログラマ必携書

IBM PC & PS/2 プログラミング 3部作



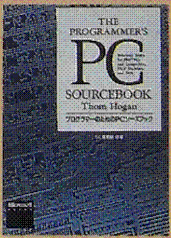
あのピーター・ノートンのベストセラー。IBMプログラマ必携のスタンダードガイダンス。

IBM PC & PS/2
プログラマーズガイド
● Peter Norton & Richard Wilton著
● 定価　5,200円



IBM PCファミリーのビデオシステム完全ガイド。

PC & PS/2 ビデオシステム
プログラマーズガイド
● Richard Wilton著
● 定価　6,000円



IBM PC & PS/2 プログラマ必携の技術資料完訳。

プログラマーのための
PC ソースブック
● Thom Hogan著
● 定価　5,200円

PC-PAGE

どれをとっても栄養満点！
おいしさいっぱいのパソコンブック

22 パソコンのお医者さん

●A5判 240頁・定価1,500円・SE編集部編

フロッピーディスクが突然読めなくなってしまった／ソフトが起動できない、ハードディスクがクラッシュした………。こうしたパソコンの「病気」に対し、内科、外科、小児科、精神科などの名医たちがトラブルを解決すべく治療のメスをふるいます。まさにパソコン版「家庭の医学」。

PC-PAGE
好評発売中！

23 ハードディスクの達人

●A5判 240頁・定価1,500円・SE編集部編

ハードディスクをフォーマットするところからはじめて、ひととおりの環境を作り上げるまでの過程をおっていきながら、そのつど必要になってくるMS-DOSの知識について丁寧に解説する、いわば自分自身の手でハードディスクをモノにするためのマニュアル。

21 フリーソフトウェア派宣言

Nifty-Serve FGALのフリーソフトウェア探索ガイド。

●定価1,500円・FGAL編

24 MS-DOS5.0 全ファイル解説

98用MS-DOS5.0の全てのファイルの意味と利用法を解説。

●定価1,500円・SE編集部編

シリーズ最新刊

1	パソコン版標準秋葉原ガイドブック	定価1200円
2	一太郎Ver.4全ファイル解説 品切れ	定価1200円
3	林檎百科	定価1200円
4	僕らのパソコン10年史	定価1200円
5	私のパソコンシステム	定価1200円
6	パソコンブックガイド1990	定価1200円
7	BBSの歩き方 品切れ	定価1200円
8	東京パソコンマップ1990	定価1200円
9	MS-DOS3.3全ファイル解説	定価1200円
10	パソコンを思想する	定価1200円
11	MS-DOSバッチファイル	定価1500円
12	プロフェッショナルのパソコン利用術	定価1200円
13	PDS白書	定価1500円
14	パソコンとMS-DOS	定価1200円
15	Cの話	定価1500円
16	パソコン見栄講座	定価1500円
17	WINDOWSがやってきた	定価1200円
18	Macintoshグラフィック工房	定価1500円
19	日本語入力読本	定価1500円
20	Windows3.0特別集中講義	定価1500円
21	フリーソフトウェア派宣言	定価1500円

SE Shoeisha Co., Ltd. 株式会社翔泳社 〒102東京都千代田区平河町2-4-14平河町KSビル TEL.03-3263-0447(代表)

翔泳社の本は全国どの書店でもお求めになれます。店頭にない場合は「至急」とお申し出の上ご注文いただければ10日位でお届けできます。

NEW DOS PROGRAMMING

特集

新しいDOS環境を探検する

MS-DOS 5.0, DOS/Vプログラミングの知識と技術

MS-DOS 5.0は、日本の多くのパソコンユーザーにとって久しぶりのメジャーバージョン更新となった。タスクスイッチャを備え、UMB/HMA/EMBを利用できるMS-DOS 5.0は、3.xxが主流だったこれまでのMS-DOS環境に変化をもたらした。また、IBM PCにおける日本語表示をソフトウェアのみで可能にしたDOS/VはDOSの新たなる可能性を示している。その中で、MS-DOSのプログラミングはどう変わっていくのだろうか。さらに、数年ぶりにバージョンアップしたMS-DOS標準アセンブラであるMASM 6.0には、高級言語のようなプログラミングを可能にする強力なマクロが備えられた。新しいDOSと新しいアセンブラを用いた、90年代のプログラミングを探ってみたい。

特集
新しいDOS環境を
探検する

MS-DOS 5.0

MS-DOS5.0の新機能とプログラミング

基本に忠実なきれいなプログラミングを！

マイクロソフト株式会社　技術支援部技術管理課　中嶋雄一

MS-DOS5.0には，さまざまな機能が新しく追加されている．これにともなって，DOSのシステムファンクションにも多くの新ファンクションが追加された．これらの新ファンクションの概要と新機能の追加によるプログラミングへの影響について，マイクロソフト株式会社の中嶋氏にお話をうかがった．

中島雄一氏

MS-DOS5.0で追加・変更されたファンクション

INT 21H システムファンクションの変更点

DOS5.0になって変更になったシステムファンクションというのは，基本的に，追加された機能に合わせてマニュアルレベルの記述が変わったというものです．そういう意味では，変更されたというよりもシステムファンクションに新しく意味が追加されたというものです．

大きなものとしては，バージョンチェックのシステムファンクションが追加されました．いままでのアプリケーションがやっていたバージョンチェックというのは，INT 21Hのファンクション30Hですが，DOS5.0で追加されたSETVERコマンドを使用するとこのファンクションで得られるバージョンはごまかされてしまいます．つまり，アプリケーションプログラムは，ユーザーがSETVERコマンドを使う限り本当のバージョンをとれないことになります．そこで，SETVERコマンドでごまかされないバージョン取得ファンクションが追加されたというわけです．

このファンクションには，MS-DOSがROMから起動したかどうかの識別機能も追加されています．

次にメモリのアロケーションに関して，UMB（アッパーメモリブロック）という概念が追加されましたから，このUMBのリンクをオン／オフするファンクションが追加されています．

また，プログラムは，アロケーションストラテジに従ってメモリを割り当てています．つまり，640KBの中で下からメモリを割り当てるか，上から割り当てるか，それともMS-DOSにまかせて最適なものを割り当てるか，ということです．その際，UMBから優先してアロケーションしていくというファンクションが追加されました．追加されたのはそのくらいです．

あと細かいところでは，IOCTLハンドルの問い合わせ，あるいはIOCTLデバイスの問い合わせという

表1　MS-DOS4.0/5.0で追加されたINT 21Hシステムファンクション

ファンクション	内容
●ファイルハンドルファンクション	
6CH	拡張オープン・作成
●ドライブ管理ファンクション	
1FH	デフォルトDPBの取得
32H	DPBの取得
3305H	起動ドライブの取得
●IOCTL（I/Oコントロール）ファンクション	
46H	メディアIDの設定
66H	メディアIDの取得
5FH	表示モードの設定
7FH	表示モードの取得
68H	メディアタイプの検出
4410H	IOCTLハンドルの問い合わせ
4411H	IOCTLデバイスの問い合わせ
●メモリ管理ファンクション	
5800H	メモリ割り当て方法の取得（UMB関係の追加）
5801H	メモリ割り当て方法の設定（UMB関係の追加）
5802H	UMBのリンク取得
5803H	UMBのリンク設定
●プログラム管理ファンクション	
4B05H	実行状態の設定
5D0AH	拡張エラーの設定
●多国語サポートファンクション	
6507H	2バイト文字テーブルの取得
●システム管理ファンクション	
3306H	MS-DOSバージョンの取得

のが追加されています．

INT 21H以外のソフトウェア割り込みの変更点

INT 21H以外のソフトウェア割り込みについては，いままでアンドキュメントだったものがドキュメント化されたというものが多いです．こういうのは，追加は追加なんですが，アンドキュメントの頃から利用されていたものが多く，ファンクション自体の中身はほとんど変わっていません．

たとえば，INT 28H MS-DOSアイドルハンドラという割り込みがありますが，これは，有名なところではPRINT.EXEが利用していました．これを利用して，スクリーンセーバーなどを作成できると思います．

このあたりのものは，たんに公開していなかっただけで，昔からあったものです．ただ，いままでは方針が不明確で次のバージョンでなくなる可能性

表2　追加されたソフトウェア割り込み

割り込み	内　容
27H	プログラムの常駐終了
28H	MS-DOS アイドルハンドラ
29H	高速コンソール
2AH	ネットワーク，クリティカルセクション
2EH	COMMAND.COM の非常駐部の再ロード
2FH	多重化割り込み
	AX =0100H PRINT.EXE が常駐しているかどうかを検査
	AX =0101H プリント待ち行列にファイルを追加
	AX =0102H プリント待ち行列からファイルを削除
	AX =0103H プリント待ち行列からすべてのファイルを削除
	AX =0104H プリントジョブを停止，プリント待ち行列のアドレス取得
	AX =0105H プリント待ち行列処理再開
	AX =0106H プリンタデバイスヘッダアドレス取得
	AX =0600H ASSIGN.COM が常駐しているかどうかを検査
	AX =1000H SHARE.EXE が常駐しているかどうかを検査
	AX =1100H ネットワークソフトウェアが常駐しているかどうかを検査
	AX =1400H NLSFUNC.EXE が常駐しているかどうかを検査
	AX =1680H プログラムの待ち状態をシステムに通知
	AX =1A00H ANSI.SYS が常駐しているかどうかを検査
	AX =4300H HIMEM.SYS が常駐しているかどうかを検査
	AX =4310H HIMEM.SYS のエントリポイントアドレスの取得
	AX =4800H DOSKEY.COM が常駐しているかどうかの検査
	AX =4810H 126文字までの行を読み込み，バッファにコピーする
	AX =4B01H クライアントプログラムの通知ファンクションハンドラリンクリストの作成
	AX =4B02H タスクスィッチャが常駐しているかどうかの検査
	AX =4B03H 一意なスイッチャ識別子を返す
	AX =4B04H 指定されたサービスファンクションハンドラを持つタスクスイッチャのスイッチャ識別子を返す
	AX =4B05H インスタンスデータの識別
	AX =0AD80H KEYB.COM のバージョン取得
	AX =0AD81H KEYB.COM のアクティブページコード設定
	AX =0AD82H KEYB.COM の国別フラグの設定
	AX =0AD83H KEYB.COM の国別フラグの取得
	AX =0B000H GRAFTABL.COM が常駐しているかどうかの検査
	AX =0B700H APPEND.EXE が常駐しているかどうかの検査
	AX =0B702H APPEND.EXE のバージョンフラグの取得
	AX =0B704H 現在追加されているディレクトリリストのアドレス取得
	AX =0B706H APPEND.EXE の動作モードの取得
	AX =0B707H APPEND.EXE の動作モードの設定
	AX =0B711H 実名フラグのセット
30H	MS-DOS エントリポイント（この割り込みを発生してはいけない）
31H	MS-DOS エントリポイント（この割り込みを発生してはいけない）

メモリ用語集
IDT（Interrupt Descriptor Table）　プロテクトモードの割り込みベクタテーブル．

があったのですが，これらが公開されたということは，今後もこのファンクションが保証されるということです．

もっとも大きな追加は，INT 2FHです．これは新しいMS-DOSの機能，たとえばDOSKEY.COMであるとかHIMEM.SYSであるとか，いままでなかったモジュールが入ってきたことに対する追加ということです．

代替ファンクションの利用

ファンクションの中に，今回から代替ファンクションというものを加えています．

これは，FCB(ファイルコントロールブロック)を使ったもの，あるいはINT 21H以外にもディスクのアブソリュートリード／ライト，INT 25H，26H，このようなファンクションを使うのをやめて欲しいということです．いわゆる，CP/Mと同じファンクションを使うのはやめて欲しいと．マニュアルには，古いファンクションを使わないで欲しいというようなコメントがついていますが，具体的にはこれらのファンクションの使用を中止して代替ファンクションとして指定されたファンクションを使用するようプログラムを変更してもらいたいということです．

いま代替ファンクションという形で古いファンクションの使用中止を呼びかけた理由は，UMBの利用であるとか，Windowsの登場などによって，あらためてこれらの古いファンクションが将来的になくなる可能性が出てきたからです．

使わないで欲しいと書かれたファンクションは，使ってはいけないと考えたほうがよいでしょう．“いけない”と明確にマニュアルに書かれていないのは，MS-DOSの以前のバージョンで動いていたプログラムは今回のバージョンでも動きますし，以前のプログラムを否定するわけにはいかないからです．ですから，MS-DOS5.0では使わないでくださいというい方になるわけです．

明確には書いてありませんが，もし“使わないで欲しい”と書かれたファンクションを使用しているとMS-DOSが次のバージョンになったとき

表3　代替ファンクション

使用中止すべきファンクション	代替ファンクション	内　容
00H	4CH	プログラムの終了
01H	3FH	ファイル，デバイスからの読み出し
02H	40H	ファイル，デバイスへの書き込み
03H	3FH	ファイル，デバイスからの読み出し
04H	40H	ファイル，デバイスへの書き込み
05H	40H	ファイル，デバイスへの書き込み
09H	40H	ファイル，デバイスへの書き込み
0AH	3FH	ファイル，デバイスからの読み出し
0FH	3DH	ハンドルによるファイルのオープン
10H	3EH	ハンドルによるファイルのクローズ
11H	4EH	最初に該当するファイル名の検索
12H	4FH	次に該当するファイル名の検索
13H	41H	ファイル削除
14H	3FH	ファイルやデバイスからの読み出し
15H	40H	ファイルやデバイスへの書き込み
16H	3CH	ハンドルによるファイルの作成
17H	56H	ファイル名変更
1BH	36H	ディスク空き領域取得
1CH	36H	ディスク空き領域取得
21H	3FH	ファイルやデバイスからの読み出し
22H	40H	ファイルやデバイスへの書き込み
23H	42H	ファイルポインタの移動
24H	42H	ファイルポインタの移動
26H	4B00H	ロードと実行
27H	3FH	ファイルやデバイスからの読み出し
	42H	ファイルポインタの移動
28H	40H	ファイルやデバイスへの書き込み
	42H	ファイルポインタの移動

メモリ用語集
アドレッシングモード　メモリアドレスの参照方法．アドレスの指定をレジスタを通さず，オペランドで直接参照する方法を直接アドレッシングモードという．

に，もしかすると苦労することになるかもしれないという意味が入っているわけですね．

いままではアンドキュメントな部分を使われて苦労したという人が多かったと思いますが，こんどはドキュメント化されているけれども苦労することになってしまうということです．ですから，ほんとうはDOS5.0で切り捨てたかった部分だけれども，準備期間をおかないと切り捨てられないので，DOS5.0はその準備期間だと考えていただきたいと思います．

MS-DOS 5.0 メモリ管理の変更について

HMA利用の問題点

UMBやHMA(ハイメモリエリア)などのメモリ管理は，XMSマネージャであるHIMEM.SYSが行っています．さらにその上で，DOSがコントロールするという形です．ただし，UMBはアプリケーションに対して使い方が解放されていますが，HMAはそうではなくDOSがにぎって離しません．握って離さないというと語弊があるかもしれませんが，DOS5.0においては，HMAはDOSのカーネルが占めてしまっていますから，ほかのプログラムがHMAに入ってくるとDOS自身が動かなくなってしまいます．また，実際にDOSカーネルがHMAに入るとHMAはほぼ満杯の状態で，ほかのプログラムを入れることができません．ですから，DOSが握ってしまうということになるわけです．

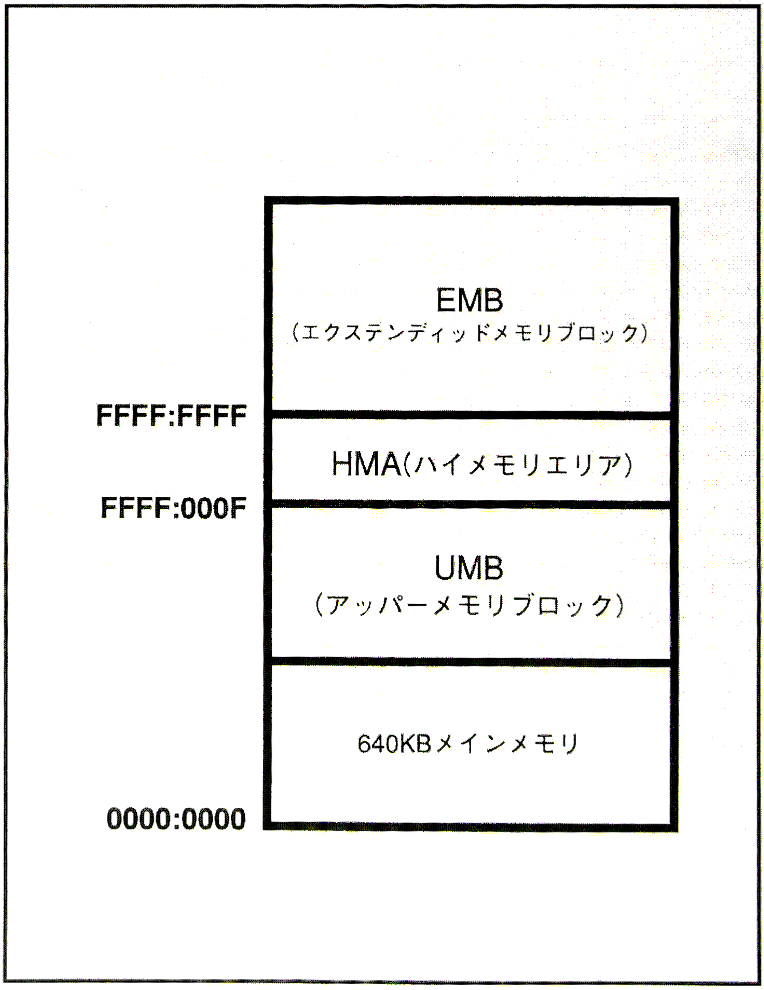
この違いだけで，UMB，HMAとも物理的なメモリはメモリマネージャが管理して，その上でDOSが使ってアプリケーションに提供しているという形をとっています．

HMAをDOSが使用していない場合には，アプリケーションがHMAを使用することができます．この場合は，XMSのファンクションにHMAが使用されているかどうかを問い合わせるものがありますので，それを利用していただくことになります．

また，HMAをアプリケーションから利用する以上，A20のオン／オフについてはつねに気を付けてプログラムを書いていただかなくてはなりません．XMSファンクションのなかにA20のグローバルとローカルのオン／オフのファンクションがありますが，それを使う際には矛盾が生じないように十分気を付けて使用する必要があります．

常駐プログラムとの併用という場合もありますので，自分自身のプログラムだけではなく，ある程度ユーザーがアプリケーションを使用する環境をも意識しなければいけなくなります．難しいとは思いますが，常駐プログラム

図1　UMB, HMA, EMB



メモリ用語集
アロケート（allocate）　EMMはアプリケーションからメモリの要求があった場合，論理ページをアロケート（割り当て）する．

自身もA20のオン／オフを意識して作られていれば大丈夫だと思います．

まぁ，A20がオンになっていれば，まずDOSが使っているだろうから，HMAは使わないというほうが無難でしょう．そういう意味では，アプリケーション側で勝手にコントロールするのはできれば避けたほうがよいのかもしれません．

また，A20がオンになっているときは，アドレスがFFFF:000FHを越えても0に戻りません．この点にひっかかるプログラムがあるようですので，やはり注意が必要です．

UMB利用の問題点

UMBについては，プログラムのサイズが問題になります．基本的にデバイスドライバなどは，最初にイニシャライズするときに必要になる領域と常駐するときに必要とする領域の2つのサイズをもっています．デバイスドライバが組み込まれる際にメモリがたりないと，普通はエラーになって組み込みが中止されます．ところが，最近のデバイスドライバのなかにはメモリがないにもかかわらず，自分で勝手にメモリがあるものとしてデータ領域を作ってしまうものがあります．このようなデバイスドライバをDEVICE-HIGHしてUMBに組み込もうとすると，ありもしないメモリにアクセスしようとして，いとも簡単にハングアップしてしまいます．

基本的には，まずMS-DOSがデバイスドライバのファイルサイズを見てUMBにロードできるかどうかを判断し，次にデバイスドライバ自身が自分をイニシャライズするのに十分なメモリがあるかどうかをチェックします．この基本ルーチンをはぶいているプログラムがあるわけです．

PC-9801の初期のタイプのように128KBしかメモリがない時代には，プログラマは当然要求するだけのメモリがあるかどうかをチェックしていたと思います．ところが，640KB実装されているのがあたりまえの時代になって，手抜きをするプログラマが出てきたというわけです．本来，自分が必要とする資源があるかどうかを確認してから動くというのが基本だと思います．初心に返るというのではありませんが，もう一度基本を見直していただきたいですね．

また，いまはデバイスドライバを例にしてお話ししましたが，メモリ常駐プログラムなどでもまったく同じ問題が起こります．

そのほかの新機能および変更点

KKCFUNC.SYS

DOS5.0では，複数のアプリケーションをタスクスイッチすることができますが，このとき問題となるのがかな漢（かな漢字変換フロントプロセッサ）です．そこで，この問題を解決するために日本語MS-DOS5.0にはKKCFUNC.SYSというデバイスドライバを追加しています．

KKCFUNC.SYSというデバイスドライバは，複数のかな漢に共通なテーブルなどを提供するもので，このデバイスドライバ自体がかな漢を切り替えることを実現しているのではありません．つまり，かな漢自身がKKCFUNC.SYSに対応してないとかな漢の切り替えは実現できないのです．

かな漢自身が共通のファンクションを使って共通のやり方でデバイス名などを登録し，切り替えのファンクションが発行されるとKKCFUNC.SYSが各かな漢に切り替えが行われることを通知してやります．各かな漢はそれを受けとって，自分がオフになるようならばフックしていたベクタテーブルを元に戻したりデバイス名を変えたりといった処理をしてオフになり，オンになるように通知されたかな漢はオンになるようにフックしようとするものをあらためてフックし，デバイス名を有効にしたりしてオンになります．

このように，KKCFUNC.SYSというデバイスドライバはあくまで仲介をするものであって，複数のかな漢を実現するのは各かな漢の問題ということです．

ディスク管理

大容量ディスクに対応するために，セクタ数を管理するデータサイズを増やしました．ディスク管理に関する大きな変更点はそこだけです．具体的には，Ver. 3.Xまでは1バイトでセクタのカウントを管理していたのをDOS 4.0から2バイトに拡張しました．

結果として，ワンパーティションとして管理できる限界が32MBから2GBに拡張しました．計算上は4GBまでいくはずが，なぜ2GBかというと，DOSのシステムファイルテーブルと

メモリ用語集
EMB（Extended Memory Block）　1MBを超えるメモリ空間のうちHMAを除く領域を指す．

図2　メモリアリーナ

```
ARENA   STRUC
        arenaSignature  db      ?
        arenaOwner      dw      ?
        arenaSize       dw      ?
        arenaReserved   db      3 dup(?)
        arenaName       db      8 dup(?)
ARENA   ENDS

フィールド       内容
arenaSignature  構造体が有効かどうかを示す
                4DH=有効
                5AH=リンクリストの最後
arenaOwner      メモリブロックの所有者
arenaSize       メモリブロックのサイズ
arenaReserved   予約
arenaName       メモリブロックを所有するプログラムのファイル名
```

いう管理領域の制約で2GBまでしかファイルの管理ができないのです．そこで，ディスクのパーティションも2GBに落としているわけです．

FATサイズが変わったんじゃないかとよくいわれますが，FATサイズは前から16ビットになっており，これは変わっていません．

メモリアリーナ

MS-DOSでは，メモリブロックの中にオーナーIDというそのメモリブロックの所有者を示す識別子のようなものをつけるのですが，その領域のことをメモリアリーナと呼んでいます．

たとえば，日本語で拡張されたものとして，ADDDRV.EXEが使っているメモリはメモリアリーナの9番ということになっています．

あるメモリブロックがどこからどこの領域を占めているかというのはわかるんですが，それをどのプログラムが使っているかというのは普通はわかりません．デバイスドライバであればデバイス名で想像することができますし，プログラムであれば起動時のファイル名で置き換えることができますが，そうでないもの，たとえばADDDRV.EXEで組み込まれたものなどはわかりません．そこで，そういうものを識別するのにメモリアリーナを使うわけです．

実際，DOS5.0に付属しているMEM.EXEなどでメモリアリーナを使っています．

これからのDOSプログラミング

DOSSHELLやWindowsなどの環境で問題なく動作するプログラムを作るには，やはり，昔からマイクロソフトがDOSを提供する側としていい続けてきたこと，ファンクションを使うべきものはきちんとファンクションを使用して，絶対番地でアクセスするようなことはしないようにする，という基本を守ってプログラミングすることが必要だと思います．いわゆる“きれいな”プログラミングをすることが大事になります．

また，マイクロソフトがこういうファンクションを使ってプログラムを作ってくださいといっているということは，逆にいうとそのファンクションを保証しなければいけないということです．ファンクションを保証するとはどういうことかというと，DOSピュアなものとDOSSHELLのタスクの上と，Windowsの各DOSボックスの上とが，アプリケーションから見てまったく同一の環境になるようにするということです．

ですから，使うべきファンクションをきちんと使って作られたプログラムは，3つの環境のどれにおいても問題なく動作すると考えていただいて結構です．

ただ，そうはいってもスピードの点，マシン固有の問題で，どうしても“きれいに”作れないということも事実だと思います．このような場合には，アプリケーションメーカーさんの判断で責任もって作っていただくしかありません．

マイクロソフトとしては，そういう作り方は勧められませんが，責任をもっていただけるのであれば，どうぞということです．

やはり，さまざまな環境，Windowsも含めて，さまざまな機種で動作するプログラムを作成するためにも，プログラマーズリファレンスに保証されているファンクションを使用し，そこからはずれないことが大事だといえます．また，そのようなプログラミングがこれからのプログラマに求められるものだと考えています．

メモリ用語集
EMS（Expanded Memory Specification）　ロータス・インテル・マイクロソフトの3社が提唱した拡張メモリの仕様．1985年に3.2版，1988年に4.0版が公開された．

特集

新しいDOS環境を

探検する

MS-DOS 5.0

アプリケーションプログラムをMS-DOS5.0に対応させる

インタビュー：株式会社アドミラルシステム　技術部第１課課長　堀　正明氏に聞く

株式会社アドミラルシステムでは，ディスク管理ツールNewton-98や，バックアップユーティリティNewton-Backupなどのアプリケーションソフトを開発，販売している．昨年11月末にNECから発売されたMS-DOS 5.0にいち早く対応し，現在では製品に「MS-DOS5.0対応」のシールを貼って販売している．同社の堀正明氏に，アプリケーションプログラムをMS-DOS5.0対応に変更する際のポイント，プログラマから見てMS-DOS5.0で変わった点についてお話をうかがった．

1パーティション2GBまで HD容量の変更への対応

――アドミラルシステムさんでは，製品にDOS5対応シールを貼っておられますね．それはいつ頃からでしょうか．

NECさんがDOS5を発表されてからです．

――そうしますと，去年の秋くらいからですか．

そうですね．ただ，アメリカではIBM版の本（MS-DOS5.0 Programmer's Reference）が出たのが去年の夏くらいですから，出て早々に手にいれまして，いろいろと使ってみました．おそらくNECさんもこれと同じ仕様だろうということで，IBMのDOS5を使って勉強はしていました．

――DOS5対応シールを貼るより前に，すでに対応は終わっていたということですね．

ええ，いくらなんでも出てすぐに対応するというわけにはいきません．もちろん，こちらの思惑とNECさんの製品が違っていては困りますから，われわれの対応の仕方でNECのDOS5に合っているかどうかを確認しました．そのへんは声を大にしていっておきたいですね．

――その辺がユーザーさんも気になるところでしょうし，開発を担当されている方々が苦労されている点だと思います．アドミラルシステムさんの製品であるNewton-BackupやNewton-98，Newton-Vなどでは，ディスクまわりの変更などは大きな関係がありそうですね．

何をもってDOS5対応とするかについては，それがなんのアプリケーションかによって大きく左右されるというのは当然の話です．Newtonなどでいいますと，やはりディスク関係ですね．ディスクまわりでの変更といいますと，1パーティションが大きくとれるようになったということですね．そのためには何をしなければならないかというと，細かい話になってきまして，クラスタやセクタの話になってくるのですが……

――FATの管理情報が変更されたと聞いていますが……

といいますか，クラスタ番号自体は変わらないのです．論理セクタ番号が，これまで2バイトだったのですが，4バイトになりました．

――基本的にはそこだけを対応すればいいのですか．

ええ，ですからそれを使っていなければ関係ありません．DOS5対応，DOS5対応といってもみなさん大きくとらえすぎではないかという感じがしますね．たとえばMS-DOSの機能を使ってファイルのオープンやリードをやっている場合には，変更する必要はまったくないのですから．FEPはまた話が別ですが，ワープロやエディタなどは動いてしまうわけですよね．

――ふつうに対応していれば動くはずだ，という話はよく聞くのですが，その割には動かないソフトも多いといいますね．

具体的にどう動かないのかを調べてみると，FEPが動かなければそれを使う必要のあるソフトは全部ダメだというのがありますね．ほかには何かありますか？

――常駐ソフトがほとんど全滅に近いですね．たとえばヒストリー関係のものなどがだめです．

そういったユーティリティ類については，DOS自身が機能を提供してしまったので，作っていた人にとってはちょっと困ったことになっているでしょうね．MEMORY-PROなどもメリットがなくなってしまいましたよね．

ディスク関係でいいますと，ふつうにファイルアクセスする分には全然関係ありません．たとえば，うちの製品の機能でいうと，ディスクの並び替え――NortonでいうとSpeed Diskにあたる機能です――などは，ファイル単位でアクセスしているわけではありません．そうしますと，小さいディスクのときは前のバージョンと互換性があるからいいのですが，大きなディスクになってしまうといままでどおりではないので，動かなくなってしまうということはあります．ディスクに関する対応というのはそのくらいですね．

――いま例にあがったNortonのSpeed Diskのようなプログラムを，2GBのハードディスクでも実行できるようにするというのは，開発者側の負担としては大きいものなのでしょうか．

他社の製品ですから(笑)どんなふうに開発しているのかにもよりますけれど，DOS5の本をきちんと読んでいれば，たいしたことではないのではないでしょうか．まず，マニュアルを読むことだと思いますよ．

――よほど奇抜なテクニックを使っていない限り対応は可能と考えていいの

メモリ用語集
EMSエミュレータ　バンクメモリやプロテクトメモリを使って疑似的にEMSメモリを実現するソフトウェア．速度的に遅くソフトウェアによっては利用できない場合もある．

でしょうか．

そうですね．DOS5でどこが変わったかということですと……マニュアルを見ても書いてないんですね．でも，たとえば，割り込みタイプの25Hのアブソリュートディスクリードや26Hのアブソリュートディスクライトなどは内容が変わっています．

タスクスイッチのファンクションはNEC版のマニュアルには書かれていない

——NECのDOS5のマニュアルには，「ここが変わった」ということは書かれていないわけですね．

書いてありませんね．「DOS3からの変更」という本だったら書いてあるのかもしれませんが，これはDOS5の本ですからね．

——アドミラルシステムさんのDOS5対応が早かったのは，やはり英語の文献で先に研究されていたからだといえるのでしょうね．

そうですね．こういう本が出ている

COLUMN

実行できないコマンド，実行すべきでないコマンド

NEC版の「MS-DOS5.0ユーザーズリファレンスマニュアル」には，第2章としてMS-DOSのコマンドがアルファベット順で記述されている．よく読んでみると，コマンドの中には「コマンドラインからの実行はできません」ないしは「単独での実行はできません」と解説されているものがある．たとえば，DOSSWAPコマンドはその機能として「DOSシェル実行時に使用されるものです．コマンドラインからの実行はできません.」と記載されている．無理に実行しようとすると

```
A>DOSSWAP

MS-DOS タスクスワッパです.

このプログラムは，MS-DOSシェルが内部的に使用するものです．コマンドラインからは実行するべきではありません．
```

と叱られることになる．

HDFORMATコマンドなども実行してみると，やはり注意を受ける．

```
A>HDFORMAT

Format Version 5.00
HDFORMAT. EXE は単独で起動できません
FORMAT. EXE を起動してください
```

単独で実行できないコマンドにエラーメッセージが出るのは当然……のはずなのだが，中にはメッセージを出さずにいきなり実行に入るものもある．MAOIXコマンドは「SETUPコマンドから利用され，アプリケーションの登録を補助する」とマニュアルには記述されているが，

```
A>MAOIX
```

と入力するだけで，いきなりアプリケーションプログラムのインストールの画面が表示されてしまう．

CHGEVになると事態はいっそう深刻だ．「SETUPコマンドで登録したアプリケーションを，DOSシェルから起動する場合に使用されるコマンドです」という記述があるのだが，単独で実行するのは絶対に避けたほうがよい．実行すると，ただちにリセットがかかる．そのうえ，カレントドライブがハードディスクの場合には，勝手にそのドライブから自動起動するよう設定を変更してしまうのである．つまり，ハードディスクにパーティションを切ってあったとしても，以後は自動的にそのドライブから起動してしまうのだ．

これを解除するには，[TAB]キーを押しながらRESETボタンを押す．こうすると，強制的に固定ディスクの起動メニューが表示されるので，そこで「自動起動」を解除すればよい．従来のバージョンでは，固定ディスクの起動メニューで「自動起動に設定する」を選択して，さらにSWITCHコマンドで起動装置をハードディスクに設定しなければ，自動起動にはならなかったのだが，MS-DOS5.0では単に固定ディスクの起動メニューで自動起動の設定を行うだけですむよう変更されている．[TAB]+RESETで固定ディスクの起動メニューを強制的に表示させることができるようにしたのも，この変更に伴う措置である．

というのを紹介してもらったので，あわててアメリカのほうから取り寄せたのです．プログラムを作る上ではこれがないと話になりませんね．

――いまはNECさんが日本語でプログラマ向けのリファレンスを出しておられますが，参考になる文献が少ないという状況はいまでもあまり変わっていないのではありませんか．

はっきりいってしまっていいのかどうかよくわかりませんが，Microsoftの本にはNECのほうに載っていないファンクションもいくつかあります．具体的にはINT 2FHというものですが，これについてはNECのほうにはあまり記述がありません．

――XMSのインターフェースとして書かれているところですね．これ以外の部分はどうなったのでしょう．

なくなってしまったわけではないでしょうが……DOS5で新しくなった機能として，よく取り上げられるDOS-SHELLというものがありますね．DOSSHELLではタスクをスワップする機能がありますが，その関係がINT 2FHなんですね．SWITH-CHERと書かれているのがタスクのスイッチのことなのですが，それに対応する記述がNECのほうにはなかったと思います．たいへんな話ではないかと私は思いますけれども．タスクスイッチというのはDOS5の目玉のひとつですよね．そして，それに対応するようにという割には，その機能が載っていないというのは不思議ですね．

――でも，DOSSHELLを使ってタスクスイッチングしているかぎりは，アプリケーションプログラム側でタスクスイッチされることを意識しなくてもいいのではないでしょうか．なにか問題が生じることはあるのでしょうか．

それが問題なのがFEPです．FEPをONにしたままタスクがスイッチされると，ほかのタスクへいったときに辞書のポインタが初期化されなくてはいけないのに，そのまま残ってしまうとなると，辞書を壊しかねないということになってしまいます．

――たとえば，アプリケーションプログラムのほうから，タスクスイッチングを禁止するようなことはできないのでしょうか．

できるはずなのですが……NECのほうにはINT 2FHのファンクションの解説がありませんから……

――たしかに，EMSが何MBあろうが，ディスクにスワップしてしまうというのが問題ではありますね．

それができるようになるのがDOS6かもしれません．

MS-DOS Programmer's Reference はプログラマの必読書

――雑誌のアンケートなどでも，アプリケーションがDOS5に対応しているかどうかをソフトハウスさんに調査をしているようなのですが，その回答というのが「現在調査中」，「対応する予定」などとかなり微妙なランクづけがされているのです．調査にそれほど時間がかかるものなのか，多少疑問に思っているのですが……

いま，本を読んでいる最中なんじゃないですか，それかリストを追いかけているとか(笑)．たとえば，そうですね，かんたんにわかるのは先ほどの話ではありませんが，Speed Diskをやってみて，壊れるかどうか試してみればいいのです．

――ほんとうに壊れていないかどうかをたしかめるのは難しいのではありませんか．

かんたんにすぐ壊れてしまうからわかりますよ．

――ほかにも，新たに追加された機能で，LOAD　HIGHやDEVICEHIGHなどUMBに組み込む機能がありますね．それも試してみるまではわからないということになるのでしょうか．

関係があるのは，常駐ソフトだけですね．それもやってみればすぐにわかるでしょうね．常駐ソフトにしても，変なことをしていなければ大丈夫なはずなのです．突然，「DOSのセグメントはここだ」と決め打ちして動作するようなことがあると，DOSがいまはHIGHメモリ領域にいく場合もありますから，そのとたんにダメになるでしょうね．ですから，きちんとした手続きをふんでいれば，問題ありません．

これまでどんな形でソフトを書いていたか，ということがDOS5になって大きく影響してきたということでしょうね．きちんとした手続きをとらないで，変な技を使っていると，いま痛い目にあっていると思いますよ．

――ところで，Newton-98でDOS5対応として変更した部分というのは，ディスク関連だけなのでしょうか．

メモリ関連の情報を表示する機能がありますので，それもなおしましたし，COMMAND.COMの作りなども全然違っているようですからそれにも対応しました．Newton-98の起動は

メモリ用語集
Windows　Microsoftが1985年にVer.1.0を発売した．1990年に発売された3.0はメモリ管理方法が大きく改善され，プロテクトメモリや仮想記憶などもサポートしている．

NT.COMとNEWTON.EXEに分かれていますから，その部分でCOMMAND.COMの構造も関係があるのです．

――変更というのは，新しく追加されたファンクションをプログラムに付加していくような形で行ったのですか．

そういうものもありますし，ディスクのAbsolute Read/Writeのように，インターフェースは同じなのに，同じ条件でパラメータが違ったら，べつのパラメータを使うようにするというものもあります．DOS5は全体としてたいへん便利になっています．いままでわからなかった情報がいろいろとわかるようになったりしています．

――やはり，マニュアルをきちんと読んでプログラムを書けばよいということでしょうね．ところで，そのきちんと読むマニュアルというのは，いったいどのマニュアルのことなのでしょうか？　これまでのお話では，必ずしもNECのプログラマーズリファレンスだけというわけにはいきそうもありませんが……

ええ，日本のマニュアルというのも，誰か日本人がアメリカから来たものを日本語に翻訳したのが元になっているはずですから，やはり英語の本を読むのが正解だと思いますよ．ただ，読む人のなかには，NECのリファレンスマニュアルを翻訳した人よりも英語のつたない人もいるかもしれませんけれど……

――いまのところは，Microsoft Pressの本がいちばん詳しい文献ということになりますか．

いや，詳しいというよりは，Microsoft自身が出しているので，信頼性が高いということなのです．DOS5だ，DOS5だといって，びっくりしないで，まず腰を落ちつけて本を読むことですね．DOS2からDOS3へ移るときだって，そういう経験をしてきているわけですから．まず，プログラマーズリファレンスに目を通すとこが肝要ではないかと思います．

MS-DOS5.0では XMSへの対応は必須

――DOS5の機能を生かすようなアプリケーションというのは，どんなものになると思いますか．

DOS5というのはユーザーさんには，たいへんメリットが大きいと思います．CONFIG.SYSの設定次第でメモリも広くとれますし．DOS5の機能を生かしたアプリケーションというと，DOSSHELLに集約されますね．SHELLを作るというのはいいアイディアだと思います．

――メモリを使って効率をあげるようなプログラムを作るにはどうしたらいいでしょうか．

Newton-98のバージョン1のときは，EMSに対応しておりまして，Newtonからアプリケーションを動かすときに，メインメモリを解放するといいますか，Newton本体で使っていたメモリを解放して，動いたらもう1度もどってきて，本体をロードするようにしていました．そのとき，ディスクから読み出したのでは遅いということで，EMSに待避してロードするようにしていたのです．バージョン2からXMSにも対応しまして，XMSがあれば，XMSのほうに本体を載せる，XMSがなければEMS，両方ともなければディスクから読むという仕様にしました．これから，DOS5やWindowsという時代になってきますと，どうしてもEMSよりXMSなんですよね．WindowsもXMSを使うし，DOSにもHIMEM.SYSが付いてくるようになりましたから，DOS5ではEMSだけの対応ではちょっとどうかなと思います．それに対応しないと，下手をするとそのアプリケーションを使うときだけは，CONFIGを変えなくてはならないということになりかねません．ひとつのCONFIG.SYSとAUTOEXECでDOS5からWindowsまでやろうとすると，アプリケーションがXMSに対応するというのはどうしても必要でしょう．

メモリ用語集
XMM（eXtended Memory Manager）　Microsoft社から提供されているHIMEM.SYSなどXMSを利用するためのソフトウェア．

COLUMN UMB，HMA，EMBを活用する──XMS規格に基づくプログラミング

XMS (eXtended Memory Specification)は，メインメモリを超えるメモリ空間にアクセスするための仕様である．MS-DOS5.0では，これらのメモリ空間を利用するためのさまざまな機能が追加されたが，XMS規格そのものは，MS-DOS5.0の登場以前から存在する．現在の規格はXMS Ver.2.0で，1988年7月にLotus/Intel/Microsoft/ASTの4社によって発表された．これは，640KB～1024KBまでの領域(UMB:Upper Memory Block)，1MBメモリを超える64KBの領域(HMA:High Memory Area)，1088KB(1024+66)超の領域(EMB:Extended Memory Block)まで含めたメモリ空間の利用方法を定めたものである（図1）．

8086CPUベースのコンピュータでは，アドレッシング空間は1MBであったが，80286，386/386SX，486/486SXCPUでは，21番目のアドレス線(A20)を有効化，無効化することによって，1MBを超える領域をアクセスすることができる．XMS規格では，この21本めのアドレス線を機種に依存しない形で利用する方法を提供し，増設したメモリをアプリケーションが確保・解放する方法を規定した．

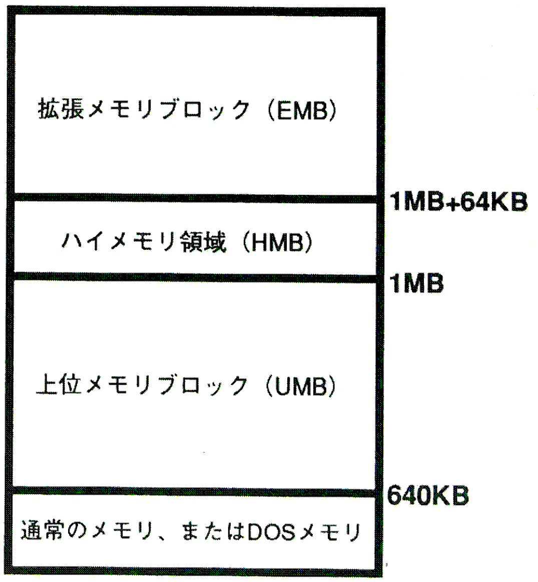
アプリケーションプログラムがXMS規格に基づいて，増設したメモリにアクセスする手順は次のとおりである．

(1) XMSドライバ(HIMEM.SYS)が組み込まれているかどうかを調べる．
(2) HIMEM.SYSのエントリ・ポイトのアドレスを取得する．
(3) AHレジスタに呼び出したいXMSのファンクションをセットする．
(4)必要に応じてほかのレジスタに値をセットする．
(5)(2)のエントリ・ポイントを呼び出す．

(1)(2)には，INT 2FHを使う．XMSドライバが組み込まれているかどうかの確認には，AX=4300Hをセットしてから，INT 2FHを実行する．XMSドライバが使用可能ならば，AL=80Hが返される．次にAX=4310Hをセットして，INT 2FHを実行すると，ES:BXにエントリ・ポイントのアドレスが返される．以後は表1に示すXMSファンクションが利用できるようになる．XMSファンクションを実行すると，成功した場合にはAX=0001H，失敗するとAX=0000H，さらにBLにはエラーコードがセットされる（表2）．

一般には，HMAを利用するプログラムが「XMS対応である」と考えられている節があるが，厳密な意味で使えばそれはHMA対応である．MS-DOS5.0では，MS-DOS自体がカーネルをHMAに置くことができるようになったため，HMAだけに対応したプログラムを利用するメリットはほとんどなくなってしまった．今後はEMBも含めた意味でXMSを利用するプログラムが求められるだろう．

図1



メモリ用語集
XMS (eXtended Memory Specification)　80286以上のCPUで1MBを超える空間にマッピングしたメモリの管理方式．HMAとEMBに分けられる．

表1　XMS ファンクション

番号	機　能
00H	XMS バージョンの取得 (呼)AH =00H (返)AX = XMS バージョン番号, BX = ドライバの内部リビジョン番号, DX =0001H(HMA が存在する) DX =0000H(HMA は存在しない)
01H	HMA の要求 (呼)AH =01H, DX =プログラムが HMA 内で必要とするメモリ容量(呼び出し側が TSR の場合)／FFFFH(呼び出し側がその他のアプリケーションプログラムの場合) (返)AX =0001H(HMA が割り当てられた) AX =0000H(割り当て失敗)
02H	HMA の解放 (呼)AH =02H (返)AX =0001H(HMA が解放された) AX =0000H(解放に失敗)
03H	A20のグローバルな有効化 (呼)AH =03H (返)AX =0001H(A20が有効化された) AX =0000H(有効化に失敗)
04H	A20のグローバルな無効化 (呼)AH =04H (返)AX =0001H(A20が無効化された) AX =0000H(無効化に失敗)
05H	A20のローカルな有効化 (呼)AH =05H (返)AX =0001H(A20が有効化された) AX =0000H(有効化に失敗)
06H	A20のローカルな無効化 (呼)AH =06H (返)AX =0001H(A20が無効化された) AX =0000H(無効化に失敗)
07H	A20の状態の取得 (呼)AH =07H (返)AX =0001H(A20が有効) AX =0000H(A20が無効)
08H	増設メモリの空き領域取得 (呼)AH =08H (返)AX =増設メモリ中最大の連続した空きメモリブロックのサイズ(KB), DX =増設メモリ中の総空き容量(KB)
09H	EMB の割り当て (呼)AH =09H, DX =要求するメモリサイズ(KB) (返)AX =0001H(メモリが割り当てられた) DX = 割り当てられたブロックのハンドル(16bit) AX =0000H (割り当て失敗),
0AH	EMB の解放 (呼)AH =0AH, DX =解放するメモリブロックのハンドル (返)AX =0001H(メモリが解放された) AX =0000H(解放に失敗)
0BH	EMB の移動 (呼)AH=0BH, DS:SI=XMM 構造体*を指すポインタ (返)AX =0001H(移動が成功した) AX =0000H(移動失敗)
0CH	EMB のロック (呼)AH =0CH, DX =ロックする EMB のハンドル (返)AX =0001H(ロックした), DX:BX =ロックされた EMB のリニアアドレス(32bit) AX =0000H(ロック失敗)
0DH	EMB のロック解除 (呼)AH =0DH, DX =ロックを解除する EMB のハンドル (返)AX =0001H(ロックを解除した) AX =0000H(ロック解除失敗)
0EH	EMB ハンドル情報の取得 (呼)AH =0EH, DX = EMB のハンドル (返)AX =0001H(ブロック情報が得られた), BH =ブロックのロックカウント, BL =システム内のフリーな EMB ハンドル数, DX =ブロックの長さ(KB) AX =0000H(情報取得に失敗)
0FH	EMB の再割り当て (呼)AH =0FH, BX = EMB の新しいサイズ(KB), DX =再割り当てするロックされていない EMB のハンドル (返)AX =0001H(再割り当てした) AX =0000H(再割り当て失敗)
10H	UMB の割り当て要求 (呼)AH=10H, DX=要求するブロックのパラグラフ数 (返)AX =0001H(UMB が割り当てられた), BX = UMB のセグメントアドレス, DX =実際に割り当てられたパラグラフ数 AX =0000H(割り当て失敗), DX =割り当て可能な最大のパラグラフ数
11H	UMB の解放 (呼)AH =11H, DX = UMB のセグメントアドレス (返)AX =0001H(UMB が解放された) AX =0000H(解放失敗)

```
* XMM構造体 ExMemMoveStruct struct
        Length          dd? ; 転送するバイト数(32bit)
        SourceHandle    dw? ; 移動元ブロックのハンドル
        SourceOffset    dd? ; 移動元ブロックの先頭からのオフセット(32bit)
        DestHandle      dw? ; 移動先ブロックのハンドル
        DestOffset      dd? ; 移動先ブロックの先頭からのオフセット(32bit)
    ExtMemMoveStruct ends ;
```

表2　XMS ファンクションのエラーコード（AX =0000H が返り，BL の最上位ビットがセットされている場合）

BL の値	意　味
80H	要求されたファンクションは存在しない
81H	VDISK ドライバが存在する
82H	A20エラー
8EH	一般的なドライバエラー
8FH	回復不能なドライバエラー
90H	HMA が存在しない
91H	HMA はすでに利用されている
92H	DX が/HMIN =パラメータより小さい
93H	HMA が割り当てられていない
94H	A20が有効になったまま
A0H	すべての増設メモリが割り当てられている
A1H	すべての増設メモリハンドルが使用されている
A2H	ハンドルが無効
A3H	SourceHandle が無効
A4H	SourceOffset が無効
A5H	DestHandle が無効
A6H	DestOffset が無効
A7H	Length が無効
A8H	移動に無効な重複部分がある
A9H	パリティーエラー
AAH	ブロックがロックされていない
ABH	ブロックがロックされている
ACH	ブロックのカウントがオーバーフロー
ADH	ロックに失敗した
B0H	指定したサイズより小さい UMB が使用可能
B1H	使用可能な UMB がない
B2H	UMB のセグメントアドレスが無効

メモリ用語集
HMA (High Memory Area)　10000Hから約64KBのメモリ空間を指す，80286以降のCPUにおいて，リアルモードから直接アクセスできる唯一のプロテクトメモリ空間．

特集
新しいDOS環境を
探検する
MS-DOS 5.0

NEC MS-DOS5.0の新機能の概要
インストールからタスクの切り換えまで

志村　拓

メインメモリの拡張，新しいビジュアルシェル，さまざまな新ユーティィリティを備えたMS-DOSの新バージョン5.0が，昨年11月末に国内でのベストセラーパソコンPC-9800シリーズ用として日本電気から発売された．ここでは新たに追加された機能を中心に，その利用方法の概要を紹介する．

MS-DOS 5.0 インストール

NECのMS-DOS5.0（以下DOS5）のディスクに収録されているプログラムのほとんどは，圧縮された形になっており，従来のように単にコピーするだけでは動作しない．この圧縮されたファイルを元の状態に復活させるためにEXPAND.EXEというファイルが入っている．自分でこのプログラムを使ってもよいが，ファイルの指定にワイルドカードが使用できないなど，何かと面倒なので，とくに理由がない限り付属のインストーラを使用して，インストールすることをお勧めする．

DOS5からハードディスクやフロッピーディスクにDOS5をインストールするためのインストーラが用意されている．これは，MAOIXを使用したスクリプトで作成されており，先の圧縮ファイルの復元や，簡単な対話形式によるCONFIG.SYSの作成といった機能もついている．もちろん，機械の作るCONFIG.SYSなんてこれっぽちも信用していはいないが，まぁ叩き台にはなるだろう．

DOS5をインストールするためには，5MB程度のディスク領域が要求される．フロッピーディスクにインストールする場合は，実にDOS5だけで1MBのディスク4枚を使用する．ぜひハードディスクが欲しいところだ．

さて，DOS5をハードディスクにインストールする際，注意と決意をしなければならないことがある．従来の3.3などのDOSでフォーマットされているディスクにもSYSコマンドを使用して，ハードディスクの領域確保などをすることなしにDOS5のシステムを転送，インストールすることができる．が，ここに大きなブービートラップが仕掛けられている．

DOS3.3までは，IO.SYSがディスク上の物理的に決められた位置に存在しなければならなかったわけだが，DOS5ではこのあたりが改善され，MSDOS.SYS，IO.SYSともにディスクのどこにあろうと立ち上がることができるようになった（ディレクトリエントリーは，先頭からIO.SYS，MSDOS.SYSの順になければならない）．このため，DOS5をインストールしたハードディスクを元のDOS3.3に戻そうとした場合にうまくシステムファイルがディスクの先頭にアロケートされず，システムを転送することができなくなるケースが出てくるわけだ．このような場合，どうしてもDOS3.3に戻したい場合は，ハードディスクのファイルをすべて退避し，ディスクの領域確保をやりなおし，システム転送を行わなければならないことになる．

MS-DOS 5.0 CONFIG.SYSを調整する

さて，うまくハードディスクにDOS5をインストールできたら，できあいのCONFIG.SYSを調整する．いかにフリーメモリを確保するかが鍵となる．とはいっても，CPUが8086やV30では何も打つ手はないし，80286でもHMSを定義することくらいしかすることがない．やはり，DOS5を楽しむには80386以上のCPUが必要だ．

CPUが80386の場合は，HMA，UMB，EMSそれにVCPIといろいろなメモリ管理を行うことができる．これらをうまく設定できればコンベンショナルメモリを600KB近く残すこ

リスト1　インストーラが作成したCONFIG.SYS

```
FILES=30
SHELL=¥COMMAND.COM /P
DEVICE=A:¥DOS¥HIMEM.SYS
DEVICE=A:¥DOS¥EMM386.EXE /P=224 /UMB
DEVICEHIGH=A:¥DOS¥SMARTDRV.SYS 1792 128
DEVICEHIGH=A:¥DOS¥PRINT.SYS /U
DEVICEHIGH=A:¥DOS¥RSDRV.SYS
DEVICEHIGH=A:¥DOS¥KKCFUNC.SYS
DEVICE=A:¥DOS¥NECAIK1.DRV
DEVICE=A:¥DOS¥NECAIK2.DRV A:NECAI.SYS
DOS=HIGH,UMB
```

メモリ用語集
エンハンスドモード　Windows 3.0において80386以上のCPUで1MBを超えるメモリ空間，仮想メモリ，複数のMS-DOSアプリケーションを起動可能にするモード．

とも夢ではない．

これらの設定を行う前に，インストーラが作成したCONFIG.SYSを見てみることにしよう（リスト1）．これは筆者の使用しているシステムで作成されたCONFIG.SYSだ．メモリ量の違いやインストールするドライブの違いから，これと同じになるとは限らないが，だいたいこれに近い形になっているハズだ．

この中でXMSの設定に関係するのは，HIMEM.SYSとEMM386.EXE（DOS5では拡張子がSYSからEXEに変わった）の2つだ．このうち，HIMEM.SYSに関しては，HMA領域の最小メモリ確保容量と，EMBのハンドル数のオプションしかなく，とくに指定する必要はない．

いかにコンベンショナルメモリのフリーエリアを増やすかの第1ステップは，EMM386.EXEの設定，その中でもとくにUMBの設定にかかっている わけだ．

UMBの活用

マニュアルを見るとUMBを使うには，EMM386に'/UMB'をつければよいとなっている．システム予約領域のうち，どのエリアが空いているかはEMM386が勝手に判断してメモリを貼ってくれるという仕様になっているわけだ．

しかし，このフルオートのUMBの設定は結構大雑把なところがあり，C0ページからDFページまでの間で8KB単位でメモリを調べ，空き領域でないとUMBとしてメモリを貼りつけないようだ．当然のことながら'/UMB'のスイッチをつけても，システ

図1　システム予約領域メモリマップ

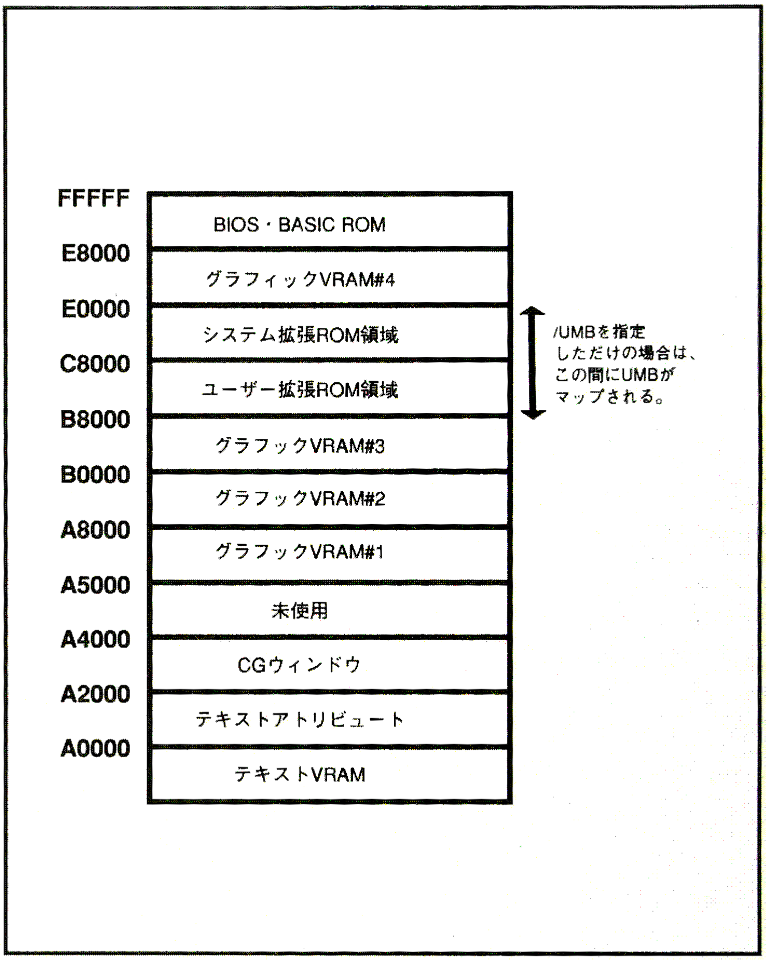


表1　おもなNEC純正ボードのROMアドレス

640KB FD I/F	D6000〜D6FFF ＊
1MB FD I/F	D7000〜D7FFF ＊
SASI HD I/F	D6000〜D7FFF
SCSI（55）I/F	DC000〜DCFFF ＊
SCSI（05）I/F	DC000〜DDFFF ＊
Sound I/F	CC000〜CDFFF
拡張 RS232C I/F	D0000〜D3FFF ＊
GP-IB I/F	D4000〜D5FFF

＊印が付いているものについては，ボード上のディップスイッチなどでROMアドレスの変更ができるようになっているので，実際のアドレスはボードのマニュアルを見ながら調べること．

メモリ用語集
OS/2　IBMとMicrosoftがプロテクトモード用OSとして1987年にリリースした．以前は次期パソコン用標準OSとして期待されていた．

ム予約領域に空き領域ができてしまうのである．

そこで，NECのDOS5では非公開オプションである'/I'オプションの登場となる．このオプションは'/E'オプションのちょうど逆の意味で，指定した範囲をUMBとして使用するというものだ．つまり，このオプションを使ってUMBの指定をすべてユーザーサイドで積極的に行ってしまおうというわけだ．

さて，どの領域をUMBに使用できるかが問題となる．賢明な読者諸氏であれば，自分のマシンのメモリマップの状態ぐらいは把握しているとは思うが，念のためにシステム予約領域のメモリマップ(図1)とNEC純正のおもな拡張ボードのROMアドレス(表1)を載せておく．

実際には，NEC純正の拡張ボードばかりを使用しているとは限らない．それらについては，ボードに付属しているマニュアルに(もしROMが搭載されていれば)ROMのアドレスが書かれているはずなので，それを参考にしながら，この際自分のマシンのメモリマップを作成してみることをお勧めする．

筆者のシステムは，PC-9801RAに拡張ボードはICM製SCSIボードが入っているだけのシンプルな構成だ．ICM製のボードはNEC純正のSCSI55ボードとコンパチで，ROMアドレスもDC000〜DCFFFとなっている．つまり，ユーザーおよびシステムROM領域で使用できないのは，このアドレスだけということになる．

試しにEMM386.EXEに'/UMB'のオプションを指定しただけの状態でDOS5を起動すると，D0000〜DBFFFとDE000〜DFFFFにUMBがマップされる(ちなみに，C0000〜CF-

表2　筆者のシステムのUMBのマップ

アドレス	内　容	EMM386まかせ	/Iで強制した結果
FFFFF			
	BIOS & BASIC ROM (96KB)		UMBをマップ？(12KB)
E8000			
	GrphVRAM #4 (32KB)		
E0000			
	未使用 (8KB)	UMBをマップ(8KB)	UMBをマップ(12KB)
DE000			
	未使用 (4KB)		
DD000			
	SCSIボードのROM (4KB)		
DC000			
	未使用 (48K)	UMBをマップ(48KB)	UMBをマップ(48KB)
D0000			
	未使用 (64K)	EMSをマップ	EMSをマップ
C0000			
	GrphVRAM #3 (32KB)		
B8000			
	GrphVRAM #2 (32KB)		
B0000			
	GrphVRAM #1 (32KB)		
A8000			
	未使用 (12KB)		UMBをマップ
A5000			
	CGウインドウ (2KB)		
A4000			
	TextAttr (4KB)		
A2000			
	TextVRAM (4KB)		
A0000			

メモリ用語集
オーバーヘッド　80286はプロテクトモードからリアルモードに戻るにはリセットしかないので命令で戻れる場合に比べオーバーヘッドが大きい．

FFFにはEMSをマップしている).ROMはDC000～DCFFFまでしか使用していないにもかかわらず,DC-000～DDFFFまでを避けてしまっていることがわかる.だからコンピュータには任せられないのだ?

そこで,'/I'オプションを利用して,キッチリ空き領域にUMBを指定することにする.ついでといっては何だが,表1を見てもわかるように,A5000～A8000にも空き領域があるので,これにもUMBをマップすることによって,メモリとして使用する.これで,標準状態よりも合計16KB余分にメモリが使用できるようになったわけだ(表2).

「たかだか16KBじゃぁ話しにならない」といった向きには,危険な香りのするUMBをマップするための領域を紹介しよう.そう,ズバリBIOS・BASIC ROM領域である.だいたいこの領域はどこまでがBIOSの領域で,どこからがBASICの領域かといったことも明確になっていないうえ,DOS自体がどのあたりを使用しているかも明確でない(すくなくとも,そうした資料をユーザーは手にすることができない)ので,割り込みベクタテーブルのセグメントアドレスや,テクニカルマニュアルなどのBIOS解説書を参考にしながら,最後はまるで実験物理(?)のように実験を繰り返し経験則を打ち立ててゆくといった状況なのである.

パソコン通信などではこの領域の使用レポートが精力的に報告されている.これらを総合的に判断し,現在筆者のシステムでは,E9000～EFFFFにUMBをマップしている.Win3の使用を諦めれば,E8000～F3FFFにUMBをマップしても問題ないようだ.ちなみに,この領域はマシンに依存するケースが多く,とくに98ノートでは,同じような設定にするとうまく動かないようだ.

EMM386.EXEでVCPIを使う

怪しげな情報ついでに,もう1つEMM386.EXEについての無責任な情報を書くことにしよう.これも,最近ではいろいろなパソコン通信などで流れているので,読者諸氏の中にもすでにご存知の向きがあるかもしれない.EMM386.EXEに1バイトのパッチ(オフセット0C686Hを00から0Dに変更する)を当てることによって,VCPI規格のファンクションが使用できるようになるというものだ*1.当然のことながら筆者の使用しているEMM386.EXEもパッチを当ててある.このパッチを当てると,'/VCPI'オプションが使用できるようになる.このオプションを指定することによって,VCPI規格のファンクションが使用できるようになるのだ.現在のところ,筆者はBCCX.EXE(ボーランドC++のプロテクトモード用ロードモジュール)を使用するくらいしか恩恵にあずかれていないが,UMBやEMSにメモリを分配した後にある程度プロテクションメモリが残るような状況であれば,試してみてはいかがだろうか?　もしかすると,この程度しか利用価値のないVCPIであれば,その分のメモリをRAMディスクとして使用した方が利用価値は高いかもしれないが.

というわけで,現在筆者のシステムのCONFIG.SYSの中のEMM386.EXEの指定はリスト2のような形で落ち着いている(EMM386.EXEではなくEMM386V.EXEになっているのは,VCPIのパッチを当てたことがわかるようにファイル名を変えたためだ).

*1　念のために断っておくが,この方法でVCPI規格のファンクションを利用することは「可能」ではあるが,保証されているわけではない.パッチ当ては個人の責任のもとで行ってほしい.

リスト2　筆者のEMM386の指定

```
Device     = A:¥DOS¥EMM386v.EXE /M=3072 /F=C000-CFFF /I=A500-A7FF /I=DD00-DFFF /I=E900-EFFF /UMB /VCPI
```


メモリ用語集
オーバーレイ　640KBのメモリ空間で,より大きなプログラムを実行するための方法.実行に必要なモジュールのみメモリにロードし,残りは順次ディスクから呼び出す.

BUFFERSとFILES

さて，CONFIG.SYSといえばおなじみのBUFFERSであるが，これが残念なことにほとんどの場合HMAに数個分程度の空きエリアしかないため，BUFFERSの値を7以上にすると，コンベンショナルメモリにその領域を確保されてしまうのである．

MemoryProやMelWareなどの従来からあるプロテクションメモリ活用ツールの中には，CONFIG.SYSの中でBUFFERSやFILESで指定するシステムの使用するバッファをUMBに追い出してしまうものが提供されていたが，残念なことにDOS5には，そうしたツールが付属していない．せっかく少しでも多くUMBを貼り付けて余計(?)なものはUMBに追い出そうと思っていても，これらのシステム使用バッファがドカンとコンベンショナルメモリに陣取ってしまっては興ざめなわけだ．

これらをUMBに追い出すツールがNIFTY-Serveにフリーウェアとしてリリースされている．これはXBUFF.COMとADDFILES.COM（両方ともK-CRAFT氏の作品）*2というもので，DOS5以上で汎用のプログラムだ．

CONFIG.SYS内のBUFFERSとFILESの指定は必要最小限にとどめ，AUTOEXEC.BATで指定したい（なぜCONFIG.SYS中でINSTALLを使用しないのかの理由については後から説明するが）．

DEVICEHIGH

DOS5から新たにデバイスドライバをUMBにロードするためのDEVICEHIGHというトークンが増えた．このDEVICEHIGHを巧く使用するには，ちょっとしたコツが要る．それは，DEVICEHIGHがUMBが寸断されている場合，その中の一番大きいブロックから使用していくというロジックをとっている点に起因する．

このため，CONFIG.SYSの中で，DEVICEHIGHは大きなデバイスドライバから順番に宣言していったほうが効率がよい．後になってくればくるほど，小さなブロックのUMBが残ってしまい，UMBにデバイスドライバがロードできなくなり，コンベンショナルメモリにロードされないようにするためだ．

INSTALL

これもDOS5(正確にはDOS4だが，NECではDOS4をリリースしなかった）から増えたトークンで，通常のロードモジュールをCONFIG.SYSの中から起動することができる．しかし，このコマンドはとくに理由がない限り使用することはない．AUTOEXEC.BATの中でLOADHIGHコマンド

＊2 XBUFF.COM及びADDFILES.COMはNIFTY-Serve FGALSTの8番ライブラリにある

リスト3　筆者のシステムのCONFIG.SYS

```
BUFFERS     = 4
FILES       = 8
SHELL       = A:¥COMMAND.COM A:¥ /P /E:640
Device      = A:¥DOS¥HiMem.Sys
Device      = A:¥DOS¥EMM386v.EXE /M=3072 /F=C000-CFFF /I=A500-A7FF /I=DD00-DFFF /I=E900-EFFF /UMB /VCPI
DeviceHigh  = A:¥DOS¥SmartDrv.Sys 1792 128
DeviceHigh  = A:¥DOS¥KKCFunc.Sys
DeviceHigh  = A:¥Bin¥DOS¥SetVer.Exe
DeviceHigh  = A:¥DOS¥Print.Sys /U
DeviceHigh  = A:¥DOS¥RsDrv.Sys
Device      = A:¥DOS¥NECAIK1.DRV
Device      = A:¥DOS¥NECAIK2.DRV A:NECAI.SYS
Device      = A:¥JFEP¥VJEB3¥VJEB.Drv /hr /sj /l /k/ /k- /k[ /kx /ks /kr /kv /d2 /DIC=A:¥JFEP¥VJEB3¥VJEB.DIC /Y128
DOS         = HIGH,UMB
```

メモリ用語集
拡張メモリ　バンクメモリやEMSメモリのように直接アドレス空間にマッピングせず，メインメモリ空間にページフレームを開き，ページ単位で割り当てアクセスする．

図2　SELKKCの実行

```
A>selkkc
 1       VJE-B Ver3.00
 2       NECAI Ver1.00

現在活動可能な 'かな漢' は 2 です.

選択したい 'かな漢' の番号を入力してください - 1
 1 が活動可能な 'かな漢' として選択されました.

A>
```

を使用したほうがコンベンショナルメモリを節約できるからだ.

日本語FEP

さて，そこで筆者のシステムのCONFIG.SYSはリスト3のようになった.「ん？」と思った方は鋭い！このCONFIG.SYSにはNEC AIかな漢字変換とVJE-βの2つ日本語FEPが登録されている. DOS5から，DOSSHELLというプログラムを使用することによって，複数のプログラムを切り替えながら実行することができるようになったのだが，この際にメモリ切り替えの都合から従来のようにADDDRV，DELDRVコマンドを使用して日本語FEPを切り替えることができなくなってしまった. そこで，DOS5では複数の日本語FEPを登録し，切り替えることができるようになっている.

この日本語FEPの切り替えを含む新しい日本語FEPのためのインターフェースがDOS5では用意された. このインターフェースがKKCFUNC.SYSである. DOS5対応，つまりこのKKCFUNC.SYSのインターフェースに対応している日本語FEPの場合は，このように単にDEVICE(またはDEVICEHIGH)で登録するだけで，FEPを切り替えることができる.

日本語FEPを切り替える命令は，SELKKC.EXEである. そう，マニュアルにろくな説明もない，あの謎のコマンドこそ日本語FEPを切り替える命令なのである.

実際に，このCONFIG.SYSで組み込まれている2つの日本語FEPをSELKKCコマンドを実行して切り替える様子を図2に示す.

AUTOEXEC.BATで常駐ソフトをUMBに

次に，AUTOEXEC.BATを見てみることにしよう. こちらはCONFIG.SYSのEMM386.EXEのように難しい点はない. ポイントは1つ，常駐ソフトの類はできる限りLOADHIGHコマンドを使用して，UMB領域に追い出してしまうようにするという点だ.

また，いくつかのコマンドで何種類かの環境変数を参照するようになったので，これらの環境変数の設定を行うとさらにGOODだ. たとえばDOSSHELLでは一時ファイルの作成先として環境変数TEMPに指定されたパスを使用するし，内部コマンドのDIRでさえ大幅に拡張されたオプションスイッチの指定方法として環境変数DIRCMDに既定値オプションスイッチを設定できるようにしている. また，その他のコマンド（CHGEV.EXE，CUSTOM.EXEやMAOIX.EXEなど）でもマニュアルには明確に書かれていないようだが，環境変数DOSDIRをDOS5がインストールされているパスとして参照しているようである. これら3つの環境変数くらいはAUTOEXEC.BATで設定しておきたいものだ（ちなみに，インストーラが作成するAUTOEXEC.BATにはDIRCMDを除いた2つの環境変数は設定されている).

続いて，AUTOEXEC.BATの中で起動しておくと便利なDOS5の常駐ソフトを3つ説明することにしよう.

便利な常駐ソフトその1 MOUSE.COM

ところで，DOS5には2つのマウスドライバが入っている. 1つはデバイスドライバの形をしているMOUSE.SYS，もう1つは常駐プログラムの形をしているMOUSE.COMだ. この2つはプログラムの形態が違うだけで

メモリ用語集
仮想EMS　仮想8086モードで80386のページング機能を使ってEMSのページマッピングをエミュレートする.

はなく，根本的に異なった2つのマウスドライバである点に注意して欲しい．

MOUSE.SYSはNEC規格のマウスドライバであり，MOUSE.COMはマイクロソフト規格のマウスドライバなのである．当然のことながら，2つのドライバを組み込んでも両方の規格が使用できるということにはならない．

ちなみに，この後説明するDOS-SHELL.EXEは，MOUSE.COMのマウスドライバでないと動かない．とりあえず，最近では（昔からという噂もあるが）NEC仕様のマウスドライバを使用するものは少ないので，MOUSE.COMを使用することにする．当然AUTOEXEC.BATの中でLOADHIGHを使用してUMBに追い払っておくことにしたい．

便利な常駐ソフトその2 MIRROR.EXE

DOS5には，1度DELコマンドなどで削除してしまったファイルを復活するUNDELETEコマンドや，誤ってフォーマットをしてしまったディスクを元に戻すUNFORMATといったコマンドが標準でサポートされた．

これらのフールプルーフのためのファイル復活プログラムはフリーウェアとしてもいくつかリリースされていた．これらは単にディレクトリエントリに残った削除ファイルの残留エントリとFATを照らし合わせて復活させるといったロジックに沿ったものだった．このため，新たにファイルを作成するなどの原因で，残留ディレクトリエントリの内容がなくなってしまうと「お手上げ」状態になってしまっていた．

MIRRORコマンドは，DOS5のフールプルーフコマンドを一層強力にするための支援常駐プログラムだ．MIRRORコマンドはDOSのFAT領域を削除関連のファンクションコールをフックして，ファイルやメモリイメージとして保存するのだ．このため，残留ディレクトリエントリの内容が上書きされることによって消えてしまっても，ファイルの復活を行うことができるわけだ．

MIRROR.EXEもとくにメモリエリアが不足しているといった状態でない限り，AUTOEXEC.BATでLOADHIGHを使ってUMBに常駐させたい．

便利な常駐ソフトその3 DOSKEY.EXE

たぶんご存知の方も多いと思うが，DOS5には，投入したコマンド履歴を編集して再発行するヒストリーと呼ばれる機能をコマンドラインに拡張するコマンドとしてDOSKEY.EXEという常駐プログラムがついている．いままでのテンプレートに比べれば雲泥の差でコマンドの投入が楽になること請け合いなのだ．DOSKEYにはヒストリー機能のほかに，あるコマンドをそのオプションと合わせて別のコマンド名をつけるエイリアス機能もついている．とくに契りを交わしたヒストリー拡張常駐ソフトや拡張シェルをおもちでない向きには，一度使ってみることをお勧めしたい．

嘘を教える SETVER.EXE

結果的には，NECではMS-DOSのバージョンが3.21（NECのいうところの3.3）から，いきなり5へのジャンプアップとなったわけだ．このおかげでいくつかのバージョンチェックを行っているプログラムが動かない場合がある．そこで登場するのがこのSETVERコマンドである．

SETVERコマンドは，指定したコマンドに嘘のバージョン番号を教えるというあこぎなプログラムである．SETVERコマンドを使用するためには，まず最初にCONFIG.SYSに

```
DEVICE=SETVER.EXE
```

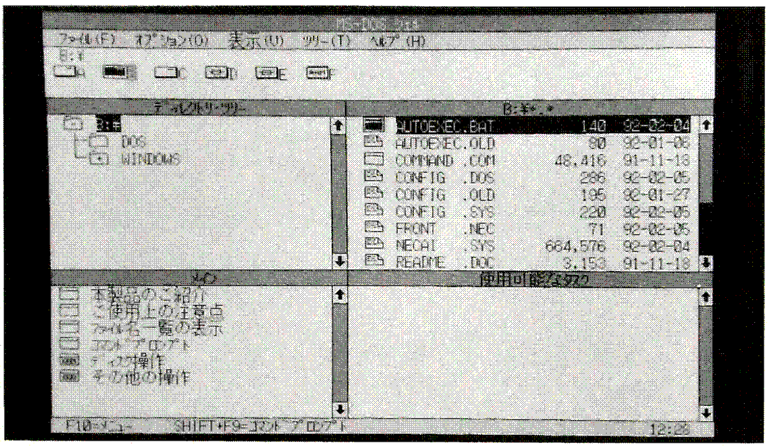
といった形でデバイスドライバとして登録しなければならない．デバイスドライバとしてSETVER.EXEが組み込めたら（つまり，1度リセットするかADDDRVで組み込む），次に

```
A>SETVER  <コマンド名>  <バージョン番号>
```

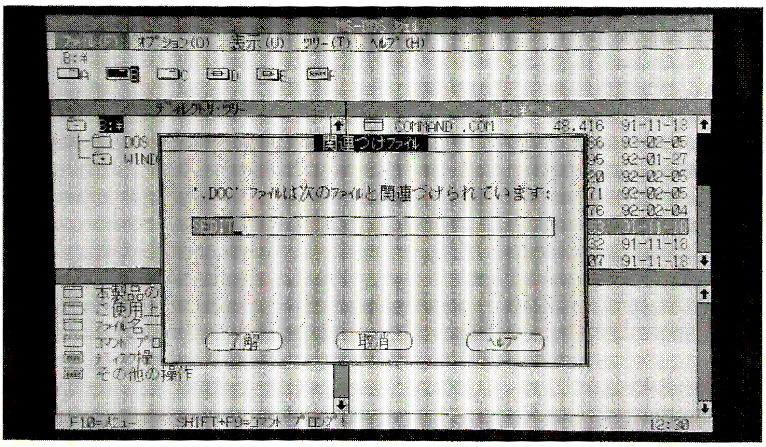
のようにして，どのコマンドに対して，バージョン番号をいくつとして嘘をつくか宣言してやる．この宣言の内容は，ただちにSETVER.EXEが自分自身を書き換えて実行ファイルの中に設けられたテーブルに書き込まれるので，1度行えば2度と行う必要はない．また，SETVERコマンドの嘘は実行ファイルだけでなく，デバイスドライバに対しても有効だ．

メモリ用語集
仮想86マネージャ　プロテクトモードでハードウェアページング機能を利用し，仮想メモリを実現するプログラム

画面１　DOSSHELL の画面



画面２　拡張子と起動プログラムの関連づけ



リスト４　DOSSHELL.INI で拡張子と起動プログラムの関連づけを行っている部分

```
        :
        :
associations =
{
    association =
    {
        program = ISH
        extension = ISH
    }
    association =
    {
        program = LHa
        extension = LZH
    }
    association =
    {
        program = VZ
        extension = TXT
    }
    association =
    {
        :
        :
```

このコマンドは考え方によっては非常に危険だ．なにせコマンドに対して嘘をつくわけなので，嘘をつかれたコマンドがうまく動かなくてもユーザーの責任なわけだ．プログラムの作者が神経質で，自分の確認したバージョンでしか動かしたくないといった理由からバージョンチェックが厳しくかかっているものに対しては，さして問題はないが，とくにFATをダイレクトに操作するようなコマンドの場合，FATのビット幅やサイズが3.21と5.0では変化しているので，これらのコマンドに対して嘘をついて無理に動かそうとすると，最悪ディスクをクラッシュさせてしまう可能性がある．

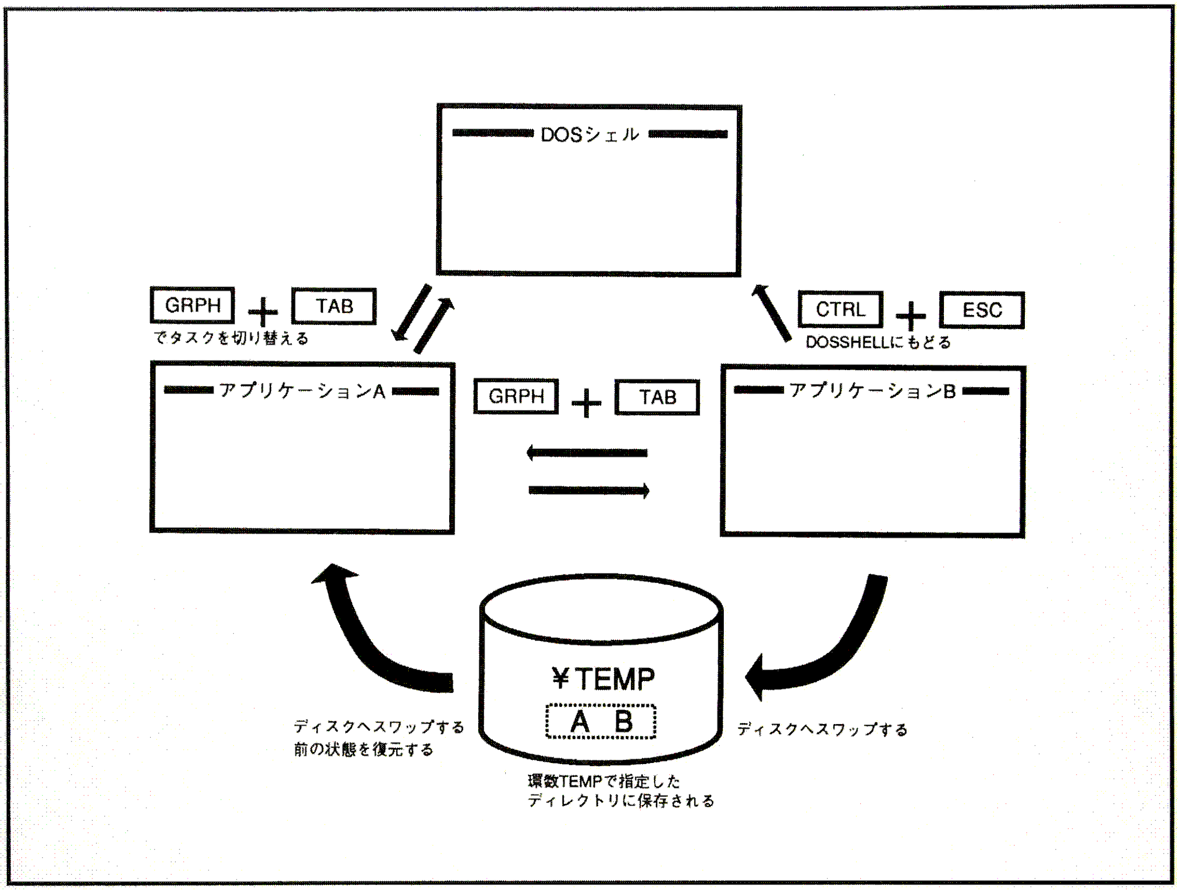
新しいDOSの顔 DOSSHELL

いままでマン・マシン・インターフェースといえばCOMMAND.COMしかなかったMS-DOSであるが，DOS5ではここにも強力なテコ入れが行われている．DOSSHELL.EXEがそれだ．これはマウスとグラフィック機能を使用したメニュードリブン型のコマンドシェルだ（画面１）．

DOSSHELLを使用すれば，CDコマンドを使用してカレントディレクトリを変更し，何度もディレクトリにあるファイルを確認するためにDIRコマンドを実行することもない．ファイルの削除，コピー，移動もマウスでファイルリストをグリグリするだけでできてしまう．よく使用するコマンドがあれば，そのコマンドをコマンドメニューに追加することもできるし，あるアプリケーションが決まった拡張子の

メモリ用語集
仮想86モード　80386以上のCPUがもつ動作モードで，仮想マシンモニタが実際のメモリ管理や入出力・割り込みを処理し，アプリケーションは8086上と同様に動作する．

図3　タスクスワップの仕組み



ファイルを作成するようなら，その拡張子と起動すべきプログラムを宣言しておくことによって，ファイルリスト中のファイル名を選択するだけで，プログラムを起動することが可能だ．DOSSHELLを使用すれば，もう，あの煩わしいキータイプ(本誌を読んでいる読者の中にキータイプが煩わしいと思っているユーザーがどれだけいるかは疑問であるが)から解放されるのだ．

プログラムリストへのプログラムの追加や，拡張子と起動プログラムの関連づけは，DOSSHELLのプルダウンメニューを使いながら対話形式で作成することができるようになっている（画面2）．しかし，この方法で1つずつ登録を行っていたのでは，さすがに精魂尽き果ててしまう．

DOSSHELLで登録したこれらの情報は，DOSSHELL.EXEの存在するディレクトリにDOSSHELL.INIというテキストファイルの形で保存される．そこで，DOSSHELLの対話形式で1つ2つ適当なサンプルを作成してから，DOSSHELL.INIをエディタで開き，サンプルのエントリーに従ってバキバキっと編集してしまう方法をお勧めする．このファイルのフォーマットは簡単なので，すぐにその内容は理解できると思うし，ヘルプメッセージなどの長い文字列を入力する場合，ダイアログボックスの中の小さな入力ウインドウで入力するより何百倍精神衛生上よい．

また，このDOSSHELL.INIの中には，「オプション」の中の「スクリーンの配色」の宣言もあるので，オリジナルの配色のエントリーを追加することも簡単に行うことができる．

タスクのスワッピング

DOSSHELLのようなビジュアルシェルならフリーソフトの類だけでもすでに相当な数がリリースされており，それぞれいろいろな特徴をもっているので，それらを使用しているという読者も少なくないだろう．しかし，

メモリ用語集
仮想86モニタ　仮想86モードを使ってEMSメモリなどを実現するためのプログラム．一般にはデバイスドライバの形式で提供される．

ここではあえて，それらの優れたビジュアルシェルからしばしの間離れてDOSSHELLを使用してみることをお勧めする．

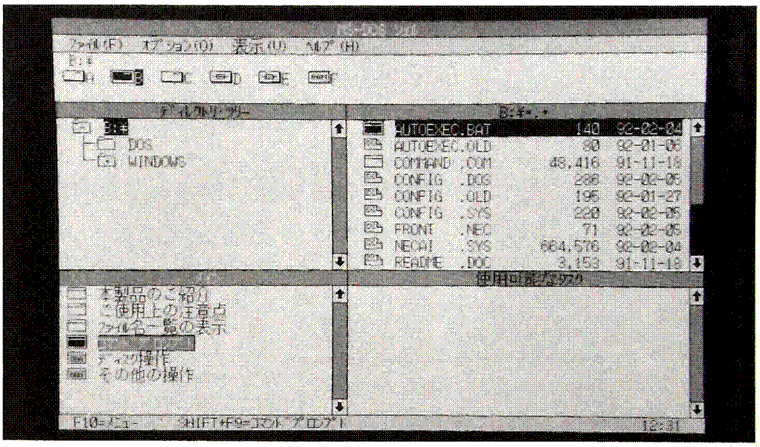
その理由は，DOSSHELLを使用することによって，タスクのスワッピングが可能になるからだ．複数のプログラムを起動して，プログラムを切り替えながら操作を行うことができるのである．

タスクのスワップを行うには，DOSSHELLを起動してから，まず最初に「オプション」の中の「タスク・スワップ・オン」をアクティブ（メニューの前に'*'が表示された状態）にする．これで，後はDOSSHELLから起動されたプログラムはスワッピンを行うことができるようになる．

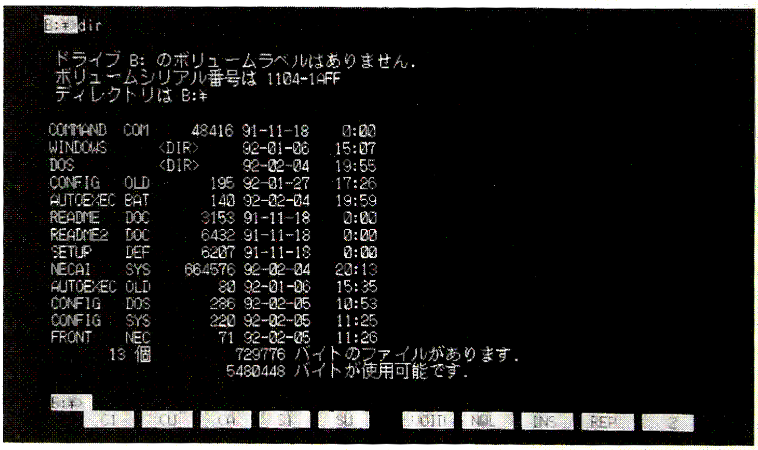
試しに，タスクスワップを行ってみよう．プログラムリストから「コマンドプロンプト」を選択してCOMMAND.COMを起動してみよう（画面3）．DOSSHELLの画面が消え，おなじみのそっけないCOMMAND.COMのプロンプトが画面に表示されているはずだ．何もしないでDOSSHELLに戻るのも何だから，DIRコマンドの1回でもコマンドを発行しておくことにしよう(画面4)．さて，この状態で[CTRL]＋[ESCキ]ーを押すと，DOSSHELLに制御が返る．(画面5）DOSSHELLが[CTRL]＋[ESC]キーを察知し，COMMAND.COMのプロセスをサスペンドし，占有していたメモリをファイルとしてディスクにスワップアウトし，制御を自分自身に戻したわけである．

DOSSHELL以外のプロセスが1つでは，なかなかイメージがつかめないと思うので，さらにこの状態からプログラムリストから「本製品のご紹介」を選択して，README.DOCを

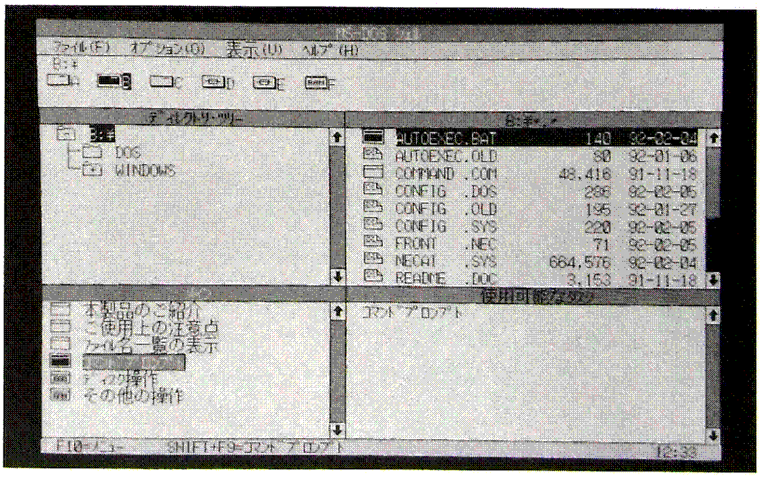
画面3　プログラムリストから「コマンドプロンプト」を選択する



画面4　DIRコマンドを実行する



画面5　[CTRL]＋[ESC]でDOSSHELLにもどる



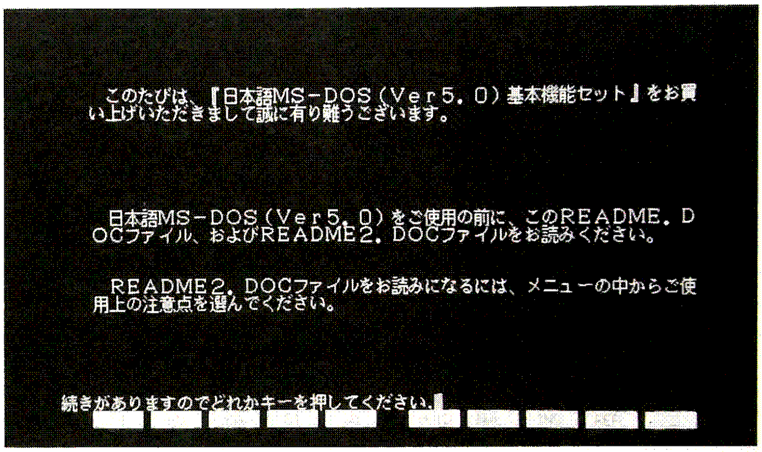
メモリ用語集
仮想記憶　実際にはマッピングされていないメモリ空間にあたかもメモリがあるようにする．実際にはメモリとディスクをスワップして実現する．

MOREさせるプロセスを起動してみよう（画面6）．画面上にREADME．DOCが表示されたら，前と同じように[CTRL]＋[ESC]キーを押してDOSSHELLに制御を戻してみよう．

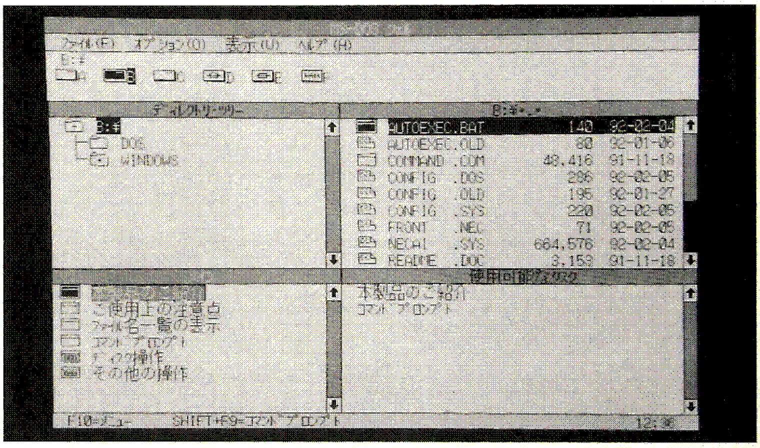
「使用可能タスク」のウインドウの中に，「コマンドプロンプト」と「本製品のご紹介」の2つアイテムが並んでいるはずだ（画面7）．試しに，「使用可能タスク」の中の「コマンドプロンプト」のほうを選択して，プロセスを再スタートさせてみよう．どうだろう，先ほどDOSSHELLに戻る前のCOMMAND.COMの状態が再現されただろうか？（画面8）

このように，DOSSHELLを使用すれば，多くのプログラムを切り替えながら使用することができるようになるわけである．先にも少し説明したとおり，各プロセスが使用していたメモリ領域のイメージは，環境変数TEMPに設定されているディレクトリに非可視属性オンのファイルの形で保存される．つまり，ディスクの領域に十分空きがあれば，相当数のプロセスを起動することができるわけだ．残念ながら，マルチプロセスではないので，タイムシェアリングしてバックグラウンドのプロセスにもCPUタイムが割り当てられるということはないが，これはこれで結構便利に使用できる．とくに環境変数TEMPでスワップアウトファイルを作成するドライブをRAMディスクなど（このためにもEMM386でVCPIを使用するのを諦め，そのメモリをRAMディスクに充てるべきか？　RAMを増設するといった解決方法も残されてはいるが……）の十分高速なディスクにしておくと，プロセスがビュンビュン切り替わって，バッチグーだ．

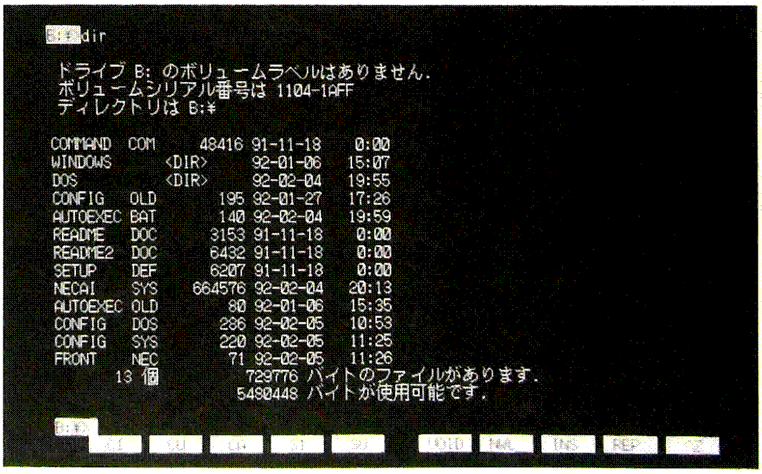
画面6　「本製品のご紹介」を選択してREADME.DOCを表示する



画面7　「コマンドプロンプト」と「本製品のご紹介」が表示される



画面8　「コマンドプロンプト」を選択すると先ほどの状態が再現される



メモリ用語集
間接アドレッシング　メモリアドレスの指定をオペランドとして指定するのではなくいったんレジスタなどに格納してから指定する方法

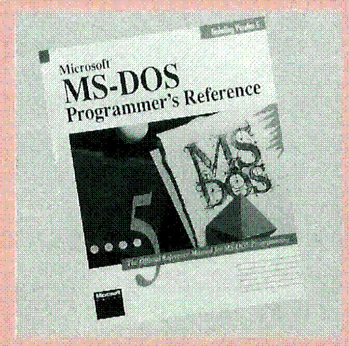
COLUMN　MS-DOS5.0 プログラミングのために――プログラマーズリファレンスを概観する

DOS5の新機能を利用する際に，プログラマにとってもっとも手近な情報源はプログラマーズリファレンスマニュアルであるが，その構成や内容は各社さまざまである．

MS-DOSは米国Microsoft社がIBM PCとその互換機用として供給しているOSで，ハードウェアメーカーはそれをそのまま，あるいはカスタマイズして，自社のマシン用に発売している．日本電気をはじめ，IBM PCを標準とする米国とは異なる規格のパソコンを製造・販売しているメーカーについては，それだけカスタマイズされている部分が多いであろうことは当然予想される．どのようにカスタマイズされたかを詳細に知ることは困難だが，各社のプログラマーズリファレンスマニュアルは，そのカスタマイズされた（あるいはされない）状態を反映した結果であるといえるだろう．

各社のマニュアルを概観することは，複数の機種に対応したプログラムを作成する場合に限らず，MS-DOS5.0本来の機能を知るうえでは意義のあることだといえよう．

Microsoft MS-DOS5.0
Programmer's Reference
Microsoft Press
29.95ドル



米Microsoft社が，MS-DOSおよびPC-DOSの5.0用として出版した公式のプログラマーズリファレンス．全体の構成は次のようになっている．

Contents
Chap.1 Introduction
Chap.2 Overview of MS-DOS
Chap.3 File System
Chap.4 Character Input and Output
Chap.5 Program Management
Chap.6 National Language Support
Chap.7 Interrupts
Chap.8 Interrupt 21h Functions
Chap.9 Device Drivers

内容はプログラマ向けの全般的な解説書となっており，とくにVer.5.0で付加された機能だけを詳しく解説したページは設けられていない．

米国では，DOS4.0からのバージョンアップにあたるため，システムファンクション（Int 21h）なども，DOSのどのバージョンからその機能が利用できるようになったかが一覧表形式で掲載されている．

Microsoft社自身が発行しているリファレンスであるため，メーカーのマニュアルでは「MS-DOSによって予約されている」としか書かれていないファンクションについても，どのように使用されているかがある程度までは記述されている．MS-DOS5.0が本来どのような機能をもつOSとして発表されているのかを知るうえでは，プログラマ必携の文献といえるだろう．

日本IBM
IBM パーソナルシステム/55
IBM DOS バージョン J5.0, J5.0/V
技術解説書
BIOS インターフェース技術解説書

プログラマーズリファレンスは「技術解説書」といういかめしいタイトルがつけられている．

内容は3部構成で，第1部がプログラム作成に関する全般的な情報（ディスクやファイルへのアクセス方法，入出力デバイスの管理，プロセス制御など），第2部がライブラリ・マネージャーやリンカ，EXE2BIN，DEBUGなどのツール類の使用方法と，ディスク・ドライバの作成について，第3部が各種割り込み，ファンクションリクエスト（Int 21h），IOCTL，EMSメモリ，タスクスワッピングなどに関するリファレンスとなっている．

「BIOSインターフェース技術解説書」のほうは，ディスプレイ入出力，ディスク，ディスケット，ASYNC，キーボード，プリンタ，タイマなど各種入出力インターフェースに関するリフ

メモリ用語集
キャッシュ　メモリを使って，HDDなどから読み込んだデータとそのポインタを記憶し，同じポインタにアクセスしたときにメモリから読み込むことで，実行速度を稼ぐ．

編集部

ァレンス，DOS/V用アプリケーションプログラムを作成するための具体的な留意事項，ハードディスクに関する情報，さらに付録としてキーボードの種類とコード，画面の定義とAPAバッファ，55xx系プリンタの状況ビットの定義などが収録されている．

日本電気
NEC パーソナルコンピュータ
PC-9800シリーズ Software library
MS-DOS™5.0 プログラマーズリファレンスマニュアル VOL.1/2

PC-9800シリーズ用のMS-DOS5.0は，アプリケーションユーザー向けの基本機能セット，プログラマ向けの拡張機能セットに分けて販売されている．拡張機能セットに含まれるのは次の5冊のマニュアルである．

ユーザーズリファレンスマニュアル
日本語入力ガイド
プログラマーズリファレンスマニュアル VOL.1
プログラマーズリファレンスマニュアル VOL.2
プログラム開発ツールマニュアル

このうち，プログラミングする上で必要となるのは，プログラマーズリファレンスマニュアル2冊と，プログラム開発ツールマニュアルである．プログラマーズリファレンスのVOL.1には，MS-DOS5.0で使用できる各種のシステムコールとファンクションリクエスト（Int 21h）のリファレンス，MS-DOS本体に関する各種の技術的な資料（ディスクのアロケーションやコントロールブロックのフォーマット，EXEファイルの構造など）が含まれる．

VOL.2のほうにはデバイスドライバのファンクション，NEC標準のかな漢字変換機能をプログラム中から利用するためのファンクション，マウスインターフェース，グラフィックスドライバ，EMS，XMS，フォントドライバのファンクションなどについての解説である．

また，日本電気のプログラマーズリファレンスは，IBM版に含まれているような開発ツールの使用方法などを「プログラム開発ツールマニュアル」として，別冊に独立させている．解説されているユーティリティは，LINK.EXE，MAPSYM.EXE，SYMDEB.EXE，LIB.EXE，MAKE.EXE，EXE2BIN.EXE，DEBUG.EXEである．

NEC版のマニュアルを，Microsoft版や日本IBM版のものと比較してみると，いくつかの違いがみられる．まず，EMS，XMSなど拡張メモリを利用するためのインターフェースが詳細に取り扱われていること，またEXEファイルの構造から，インテルのオブジェクトモジュールフォーマット，アスキーコード表，エスケープシーケンス表，UMBのメモリマップまで記載されており，プログラマにとって必要な情報は盛りだくさんである．しかも，ファンクションのリファレンスについては，3.3のときよりレイアウトが改善されて，非常にみやすくなっている．ただし，MS-DOS5.0で新たに導入されたタスクスイッチの制御ファンクションなど，Int 2Fh関係の機能については，XMSの一部をのぞいてほとんど記述がない．

富士通
日本語MS-DOS™V5.0
プログラマーズリファレンス
FM-R80/70/60/50,
FM TOWNS

富士通版MS-DOS5.0もNEC版と同じくユーザー向けの基本機能，プログラマ向けの拡張機能の2つのセットがある．拡張機能セットには，以下の3冊のマニュアルが含まれている．

MS-DOS5.0アドバンスドガイド
プログラム開発ツールリファレンス
プログラマーズリファレンス

アドバンスドガイドはユーザーが機器構成に合わせた設定を行うためのガイドブック，開発ツールリファレンスはリンカやMAKEなどのツールの解説書である．注目したいのは，プログラマーズリファレンスだ．富士通版は，構成・内容ともMicrosoft Press版のProgrammer's Referenceとそっくりである．英語は苦手だが，Microsoft Press版を読みたいという人は，FMRやTOWNSのユーザーでなくとも1冊持っていると便利だろう．

メモリ用語集
CPUキャッシュ　実行速度の速いCPUにおいてはDRAMアクセスにもウェイトがかかる．そこで高速なSRAMを用意し，キャッシングさせて実行速度を稼ぐ．

特集

新しいDOS環境を

探検する

DOS/V

DOS/Vとはなにか?

DOS/Vプログラミングの基礎を探る

柳澤健二

最近のパソコン業界にホットな話題を提供しているDOS/V．プログラマの多くも，この世界標準パソコンでのプログラム開発に興味をもっていることだろう．しかし，現実にはIBM PCおよびDOS/Vにおける技術情報は，ほとんど提供されていない．そこで，このDOS/Vにおけるプログラミングの基本について，とくに英語モードとの違いを示しながら解説した．

DOS/V イントロダクション

Windows, DOS5.0, OS/2といったコンピュータ界のホットな話題, さらには日米経済摩擦解消という心理的追い風もあってか, すっかり国内パソコン界の“台風の目”的存在になりつつあるDOS/V.

しかし「PC-9801の牙城は崩れるか」みたいな視点の記事は別として, プログラマに直接役立つ情報などはまだ不足ぎみかも……いや, ほんとうはその気になれば情報は手に入るけれど, まだあまりいき渡ってない, と表現したほうが正確かもしれません.

DOS/Vは, ひとことでいえば,

“ガイコク人専用と思われていた国際標準パソコン＝IBM　PC/ATまたはPS/2またはそれらの互換機(以下PCと略)でそのまま動作する日本語DOS”

です. したがって日本語関連部分を別にすれば, 海外のPCソフトや関連情報などがそのまま役に立ちます. 技術解説書は原書で読まなくても翔泳社からいろいろ邦訳が出てますしネ.

おまけにMS-DOSの一種ですから, たとえばPC-9801のMS-DOS上でのプログラミングとも基本部分に大きな違いはありません.

そう, あなたにDOS上のプログラミング経験があるなら, DOS/Vなんて全然こわくないっ!

といいつつ, 半面, 親戚筋のAXやJ-3100, PS/55高解像度DOSなどとも事情が違う部分はありますし, DOS/V自体「日本語」と「英語」という異質な環境を含んでいます. とくにこの「日本語環境」についての情報源は, たしかにいまひとつ十分じゃないかもしれません.

というわけで, 本稿ではその“日本語環境”を中心に, DOS/V上のプログラミングのための基礎知識, 注意点, その他もろもろ雑多な情報を, 筆者の手に負える範囲で概説します. 読者の皆さんのDOS/V活用の手助けになれば幸いです.

なお, サンプルソースリストはTurbo Pascal Version 6.0 (IBM版) 用になっています. 実際にこのソースを流用される方は, 頭のほうにこう宣言してください.

```
uses DOS;
var Regs:registers;
```

DOS/Vの画面表示のしくみ

リスト1は, IBM DOS5.0/Vの日本語モード用CONFIG.SYSの一例です.

「ハード側にあらかじめ用意しなくても, ソフトの力だけで日本語が取り扱える」

というDOS/Vの仕組みは, このCONFIG.SYS中でインストールされる$FONT.SYSと$DISP.SYSという2つのデバイスドライバに集約されています.

$FONT.SYS

$FONT.SYSは, 必要に応じて表1のようなディスク上のフォントファイルからフォントパターンを読み出し, PCの拡張メモリ領域に格納します.

もちろん, ロードされたフォントをDOSやユーザーのプログラムの要求に応じて実際に提供する機能も, $FONT.SYSが受けもっています.

$DISP.SYS

VGAを搭載したPCでは640×480ドット16色のグラフィック表示が可能です. $DISP.SYSはこのグラフィック画面を用い, 表2のような5つのビデオモードで日本語表示機能が提供できるよう, PCのBIOSの機能を拡張します.

PC本来のビデオサブシステムでは, テキストモードではセグメントB800H, 640×480ドット16色グラフィックのモードではA000Hからはじまるアドレスのメモリにビデオバッフ

リスト1　DOS/V用のCONFIG.SYSサンプル

```
BUFFERS   =20
FILES     =20
DOS       =HIGH,UMB
COUNTRY   =081,932,C:¥DOS¥COUNTRY.SYS
SHELL     =C:¥DOS¥COMMAND.COM /P /E:384
DEVICE    =C:¥DOS¥$FONT.SYS /24=ON /P=C:¥SYS¥ /U=188
DEVICE    =C:¥DOS¥HIMEM.SYS
DEVICE    =C:¥DOS¥EMM386.EXE 1024 RAM
DEVICEHIGH=C:¥DOSV¥$DISP.SYS /MSG=OFF
DEVICEHIGH=C:¥DOSV¥$IAS.SYS /R=1
DEVICEHIGH=C:¥DOSV¥$PRNESCP.SYS
```

メモリ用語集
シャドウRAM　実行速度の遅いCPUのROMアクセス時にはウェイトがかかるので同じアドレスをもつシャドウRAMにROMの内容をコピーし, 実行速度を稼ぐ.

ァがあります．

ところが日本語モードでは，テキストモードでも$DISP.SYSのはたらきによってモードがすり替えられ，実際にはグラフィックのビデオバッファが使われます．

このためDOS/V上での日本語表示は，直接または間接的にBIOSの割り込みを用いて文字を書き込んだときだけ有効です．

テキスト画面を使用する英語モード用のプログラムには，表示の高速化のためB800Hからはじまるビデオバッファに直接データを書き込むものが多いのですが，上のような理屈で，こうしたプログラムは結果的に「暴走する恐れはまずないけれど画面に何も表示されない」という状態におちいることになります．

論理的には「テキストビデオバッファをつねに監視し，アプリケーションが何か書き込んできたらそのデータをBIOSに送りなおしてやれば解決できる」わけで，実際そういうはたらきをするTSRがフリーソフトウェアやシェアウェアとしていくつか流通していますが，標準の$DISP.SYSではサポートされていません．

DBCS/SBCS

ところで「日本語モード」,「英語モード」というほうがとおりがよいので本稿でもそうしていますが，DOS/Vの「英語モード」は，実際にはドイツ語，フランス語，スペイン語など，アルファベット圏に属する世界各国語がサポート可能ですし，中国語，韓国語などを扱うことも「日本語モード」と同じ仕組みで可能で，すでにDOS/V中国語版というのもあるそうです．

そんなわけで，広い意味での“文字の取扱い方の側面での環境の違い”を表現するときには，SBCS（Single Byte Character Set＝1バイト文字セット），DBCS（Double Bytes Character Set＝2バイト文字セット）といういい方が用いられます．

日／英の環境の判定方法

DOS/VではCHEV.EXEやSWITCH.BATというコマンドで，ユーザーが自由に日／英（DBCS／SBCS)環境を行き来できます．したがって，どちらか一方での動作しか想定しないソフト，あるいは両方で使えるけれど細かい動作調整が必要なソフト……いずれにしても現在どちらの環境でプログラムが実行されているかは確認したほうがよいでしょう．

具体的には，次のようにDOSの「DBCSベクタの取得」サービス（AX＝6300H INT 21H）を利用します．

返ってきた“DBCSベクタ”の最初の1ワードが0ならSBCS，そうでなければDBCS環境と判定できます．

表1　$FONT.SYSがロードするフォントファイル

$JPNHN16.FNT	8×16ドット半角	＊1
$JPNHN19.FNT	8×19ドット半角	＊1
$JPNZN16.FNT	16×16ドット全角	＊1
$JPNZN24.FNT	24×24ドット拡張漢字	＊2
$SYSIZ24.FNT	24×24ドットユーザー外字	＊3
$SYSIZ16.FNT	16×16ドットユーザー外字	＊3

＊1　漢字ROMを装備している機種では不要だが，/F＝HN　/F＝ZNというパラメータで強制的にファイルから読み込ませることもできる．

＊2　$FONT.SYSのパラメータに/24＝ON（IBM拡張漢字など24ドットフォントの一部または全部？をROMにもたないプリンタのサポート用）を指定したとき必要．

＊3　/U＝0（外字を使用しない）を指定したときは不要．外字は最大1880まで94個単位で使用量を指定できる．

表2　日本語がサポートされるビデオモード

モード (HEX)	モード名	表示文字数	文字のサイズ 全角	文字のサイズ 半角	解像度 DBCS環境	解像度 SBCS環境
03	16色文字モード(エミュレートCGAテキスト)	80×25	16×19	8×19	640×475	720×400
11	2色カラーグラフィック	80×30	16×16	8×16	640×480	640×480
12	16色カラーグラフィック	80×30	16×16	8×16	640×480	640×480
72	16色カラーグラフィック	80×25	16×19	8×19	640×480	--------
73	16色文字モード(拡張エミュレートCGAテキスト)	80×25	16×19	8×19	640×475	--------

メモリ用語集
スタック　一時的にデータを格納するための記憶領域．関数や手続き処理において引数やアドレスを格納するのに使われる．

ただし，これはPS/55の高解像度日本語DOS（以下JDOSと略）と共通の方法です．「プログラムがDOS/V上で動いているかどうか」までチェックするには別のBIOSサービスを併用する必要があります．全部書くと長くなるので，興味のある方は日本IBMの『IBM DOS J5.0/V BIOSインターフェース技術解説書』を参照してください．

各画面モードと日本語表示

英語モードのBIOSでは，表3に示すビデオモードがサポートされています．日本語モードについてはすでに表2に紹介しました．いずれの場合にも，ディスプレイをこれらのビデオモードにセットするには，リスト3のようにAH=0 INT 10HのBIOSサービスを用います．

実際にサポートされていないモード番号にセットした場合もエラーコード等は返りませんので，場合によってはAH=0FH INT 10Hを用いて，現在のビデオモードを判定しなおす必要があるかもしれません（リスト4）．

また，実際にセットしたいビデオモードに80Hを加えた値を用いると，現在画面に表示中のデータをクリアせずにビデオモードを切り替えることが可能です．いい換えれば，80H以上のビデオモード番号は，通常，最上位ビットを切捨てて解釈する必要があります．

日本語モードでは，表2以外のモード番号については公式には「予約ずみ」となっていますが，実際はSBCS環境でサポートされている画面モードはいずれも設定可能（素通し）で，単に日本語の表示が行われない（見た目は英語環境と同等になる）だけです．ただし，日本語関係のデバイスドライバやTSRが生きているわけですから，あまり安定した動作は望めないかもしれません．

モード03H

日本語モード始動直後の標準設定で，日常的にも使用頻度が高いのがモード03Hです．これは英語モードで初代IBM PC以来もっともポピュラー

リスト2　DBCS環境のチェックルーチン

```
function IsDBCS:boolean;
{現在の環境がDBCSならTRUE、SBCSならFALSEを返す関数}
begin
    Regs.AX:=$6300;
    MsDos(Regs);
    IsDBCS:=(MemW[DS:SI]<>0)
end;
```

表3　VGAでサポートされるSBCSビデオモード

モード (HEX)	タイプ	表示 文字数	グラフィック 解像度	色数	旧ビデオアダプタとの互換
00,01	テキスト	40×25		16	CGA EGA MCGA
02,03	テキスト	80×25＊		16	CGA EGA MCGA
04,05	グラフィック	40×25	320×200	4	CGA EGA MCGA
06	グラフィック	40×25	640×200	2	CGA EGA MCGA
07	テキスト	80×25		MONO	MDA EGA
0D	グラフィック	40×25	320×200	16	EGA
0E	グラフィック	80×25	640×200	16	EGA
0F	グラフィック	80×25	640×350	MONO	EGA
10	グラフィック	80×25	640×350	16	EGA
11	グラフィック	80×30	640×480	2	MCGA
12	グラフィック	80×30	640×480	16	
13	グラフィック	40×25	320×200	256	MCGA

＊CRTコントローラの設定によって43行／50行表示も可能．

リスト3　ビデオモードの設定

```
prodedure SetVideoMode(Mode:byte);
begin
  with Regs do begin
    AL:=Mode
    AH:=$00;
    Intr($10,Regs)
  end;
end;
```

メモリ用語集
スタンダードモード　Windows 3.0において80286以降のCPUで1MBを超えるメモリ空間を利用可能なモード．

なCGA互換の80×25行16色テキストモードを，ハードウェアではなくBIOSのレベルでエミュレートするものです.

つまり「英語モード上で，画面表示はすべて表4にあげるINT 10H（ディスプレイ入出力）のBIOS CALLを使って行うようにした状態」と同じと考えてプログラミングすればほぼOKですから，PCのビデオBIOSの解説書などはすべてそのまま参考書として役に立ちます.

ただし，いくつかの点で，英語モードとの事情の違いを考慮する必要があります.

・英語モードでは複数もつことが可能な「ビデオページ」は，日本語モードでは常にページ番号0のみが有効.

・背景色の属性に「ブリンク」を指定した場合，高輝度色で代用される.

・画面の最下行は，通常，システムが日本語入力・キーボードシフト状態の表示域として使用する.

・1バイト文字の割り当てが，アルファベットとおもな記号以外大幅に異なるため，罫線のキャラクタなどを用いて画面を構成する場合，日／英のモードごとに表示用キャラクタコードの差し替えが必要（『DOS5.0/Vユーザーズ・ガイド』付録E「コード・ページ表」参照）.

また，VGA搭載機の英語モードでは，同じ「モード03H」でも画面の表示行数に25，43，50行と，3つの可能性があります．一方，日本語モードでは80×25行固定ですから，そういう意味ではあくまでも「CGA互換」であることを意識する必要もあります.

なお，この画面の実際の解像度は640×480ドット16色．高さ16ドットの文字を25行並べると行間が3ラインあきます.

DOS/Vの半角フォントが2種類あるのは，高さ16ドットのフォントで行間をそのままあけておいたのでは罫線キャラクタ表示などが途切れて不都合なためで，03H，72H，73H（つまり25行表示のモード）では，$JPNHN19.FNTに含まれる高さ19ドットの半角フォントが使用されます．もう1つの$JPNHN16.FNTは，モード11H，12Hの30行表示（行間ゼロ）用です.

疑似ビデオバッファ

表4の最後にあるAH=FEH，AH=FFHのサービスについて補足します.

先ほど触れたとおり，画面の高速化のためにビデオバッファに直接文字・属性を書き込んでいるソフトはそのままでは画面表示ができませんが，そうした表示ルーチンをもつソフトを，ソースレベルで日本語表示対応に変更するのは実は簡単です.

というのは，モード03Hに限っては「疑似ビデオバッファ」と呼ばれる領域がBIOSによって用意されているためです.

・従来のやり方

①B800:0000Hから起算して行・桁に対応するアドレスを計算.

②その位置に，1文字あたり2バイト（キャラクタコード+属性）のデータを書き込む.

・疑似ビデオバッファ書き込みのやり方

①疑似ビデオバッファのアドレスを取得する（AH=FEH）.

②そのアドレスから起算して行・桁に対応するアドレスを計算.

③その位置に1文字あたり2バイトのデータを書き込む（更新する）.

④「画面表示の更新」のBIOS割り込みを実行する（AH=FFH）.

つまり②と③の部分はほぼそのままで，

A）B800:0000Hに固定されていた書き込み位置を可変にし，

B）事前にその実際のアドレスを取得し，

C）書き込んだ直後には「どこから何文字更新したか」をシステムに通知する.

という手順をつけ加えれば，比較的高速に画面が更新できるようになっています.

この2つのファンクションは，英語モードの動作には影響を与えないた

リスト4　現在のディスプレイ状況を得る

```
function CurrentVideoMode:byte;
begin
  with Regs do begin
    AH:=$0F;
    Intr($10,Regs);
    CurrentVideoMode:=AL;
    {この他、画面の桁数がAHに、ページ番号がBHに返ります。
     日本語モードではBHはつねに0です}
  end;
end;
```


メモリ用語集
スワップ（Swap）　メインメモリと拡張メモリ，メインメモリとディスクなどの間でデータやプログラムの交換をすること.

表4 INT10Hのビデオ BIOS（日本語モード）

「AHの値」で抜けている数値は日本語モードでは「予約済み」となっている.

AHの値	機能内容	呼び出し時の設定	レジスタに返される値	備考
00H	ビデオモードの設定	AL＝ビデオモード番号	なし	
01H	カーソル形状の設定	CH＝カーソル上部の走査線番号 CL＝カーソル下部の走査線番号	なし	テキスト画面のみ
02H	カーソル位置の設定	BH＝ページ番号（0に固定） DH＝行（0～） DL＝桁（0～）	なし	
03H	カーソルの状態（位置と形状）の取得	BH＝ページ番号（0に固定）	CH＝いちばん上の走査線番号 CL＝いちばん下の走査線番号	
05H	活動ページの選択	AL＝活動ページ番号（0に固定）	なし	実用性0
06H	画面のスクロールアップ……画面の一部（ウィンドウ）単位でのスクロールが可能.	AL＝スクロール行数 BH＝新たなブランク行の表示属性 CH＝左上隅の行位置 CL＝左上隅の桁位置 DH＝右下隅の行位置 DL＝右下隅の桁位置 （BH＝0でウィンドウ内クリア）	なし	
07H	画面のスクロールダウン	06Hと同様	なし	
08H	カーソル位置の文字コードと表示属性値の取得	BH＝ページ番号（0に固定）	AH＝属性値（テキスト画面のみ） AL＝文字コード	
09H	現在のカーソル位置への文字と属性の書き込み	AL＝文字コード BH＝ページ番号（0に固定） BL＝属性値（テキスト） 前景色（グラフィック） CX＝繰り返す文字数	なし	
0AH	現在のカーソル位置への文字の属性なし書き込み	AL＝文字コード BH＝ページ番号（0に固定） CX＝繰り返す文字数	なし	テキスト画面用
0CH	ドットの書き込み	AL＝カラー値（パレット番号） BH＝ページ番号（0に固定） DX＝Y座標（0～） CX＝X座標（0～）	なし	グラフィックのみ
0DH	ドットの読み取り（グラフィック）	BH＝ページ番号（0に固定） DX＝Y座標（0～） CX＝X座標（0～）	AL＝カラー値（パレット番号）	グラフィックのみ
0EH	テレタイプ式書き込み	AL＝文字コード BL＝前景色(グラフィックの場合)	なし	
0FH	ディスプレイ状況	なし	AL＝現在のビデオモード番号 AH＝1行の桁数 BH＝ページ番号（常に0）	

メモリ用語集
セグメント　1MBのメモリ空間を16ビットのレジスタで操作するために8086の実アドレスはオフセットとセグメントを4ビット左にシフトさせた値との和で決定される.

10H	カラーパレットインターフェース	ALにサブサービス番号を指定することにより，カラーパレット設定を制御する10の機能が利用できる．（以下，一部抜粋） AL＝00H　パレット個別設定 BL＝パレットレジスタ番号(0..15) BH＝カラー・レジスタ値(0..63) AL＝07H　パレット個別読み取り BL＝パレットレジスタ番号　→BH＝カラー・レジスタ値 AL＝10H　カラーレジスタ個別設定（VGAの機能） BX＝変更するカラーレジスタ番号 DH＝R（赤）の値 CH＝G（緑）の値 CL＝B（青）の値 AL＝15H　カラーレジスタ個別読み取り BX＝変更するカラーレジスタ番号　→DH＝R（赤）の値 CH＝G（緑）の値 CL＝B（青）の値		
11H	キャラクタジェネレータインターフェース	1バイト文字セットの設定 AL＝00H（サブサービス番号） ES：BP＝フォント記憶域ポインタ CX＝設定する文字数 DX＝最初の文字のキャラクタコード BL＝設定するブロック（0に設定する） BH＝文字当りのバイト数		EGA互換
13H	文字列の一括読み込み，書き込み	ES：BP＝文字列のポインタ CX＝文字列の長さ DX＝読み書きする位置（Y，X） BH＝ページ番号（0に固定） AL＝00H・01H　文字列書き込み． BL＝属性 …単一の属性で書き込み．01Hの場合カーソルが移動する． AL＝02H・03H　文字列書き込み． …属性つき書き込み．1文字単位の属性情報つき文字列データ（文字コード，属性，文字コード…）を用意しておく． AL＝10H・11H　文字ブロック読み込み． …カーソル移動なし文字ブロック読み込み．11Hはモード73H専用で，文字コードのあとに3バイトの属性値が続く． AL＝20H・21H　文字ブロック書き込み． ……20H＝「文字コード，属性，文字コード」形式 ……21H＝「文字コード，属性0，1，2，文字コード」形式		
18H	文字フォントパターンの要求と変更	AL＝00H　読み取り ＝01H　書き込み（外字用） BH＝0（固定） BL＝文字セット（0に固定）	AL＝00H　正常終了 それ以外　エラー	

メモリ用語集
セレクタ　プロテクトモード時セグメントレジスタはセレクタとして利用され，ディスクリプタテーブルへのインデックスとして利用される．

		CH＝全角第1バイト／半角なら0 CL＝全角第2バイト／半角コード DX＝文字の幅，高さ 0810H＝8×16ドット半角 0813H＝8×19ドット半角 1010H＝16×16ドット全角 1818H＝24×24ドット全角 ES：DI＝文字パターンのバッファの先頭アドレス		
1AH	ディスプレイ組み合わせコード（コンビネーションコード）読み取りと設定	AL＝00H ……接続されているビデオサブシステムとディスプレイの種別を得る．	AL＝1AH（この機能がサポートされている） BL＝00H　ディスプレイなし ＝07H　モノクロVGA ＝08H　カラーVGA	
1DH	キーボード・シフト標識域の表示と消去	AL＝00H　表示 AL＝01H　消去 AL＝02H　状況読み取り	BX＝標識域の行数	
FEH	疑似ビデオバッファアドレス取得	ES：DI＝B800：0000H	ES：DI＝実際のバッファのアドレス（疑似ビデオバッファがないなら値は変化しない）	モード03H専用
FFH	画面表示の更新	ES：DI＝更新するビデオバッファの先頭アドレス CX＝文字数		モード03H専用

め，一度このような手順に変更してしまえば，どちらのモードでも問題なく動作します．

この手法はDOS/Vの独創ではなく，アメリカのIBMがかつて開発していたTopView（Windowsの親戚．現在のWindowsにある「PIFファイル」はTopViewから受け継いだもの）のインターフェース仕様を流用したものです．

このため，TopViewに対応している英語モード用のソフトはそのままDOS/V日本語モードでもほぼ問題なく動作します．たとえば，Norton Utilities 6.0やNorton Commander 3.0などがその例です．

モード73H

日本語モード固有のビデオモードで，1つの文字に対して指定できる属性が0，1，2の合計3バイトあります．

属性バイト0は，前景・背景各16色の色属性を設定するもので，モード03Hの1バイトの属性と意味は同じです．

属性バイト1では，いまのところアンダーラインと縦横の罫線がサポートされています．

属性バイト2はすべて「予約ずみ」で，0に設定します．

というわけで，表計算ソフトやワープロなどの画面表示に便利なビデオモードといえるでしょう．なお属性バイト1と2は，AH＝13Hの「文字ブロックの読み書き」のときだけ参照・指定できるものです．それ以外の割り込みを使用して，属性バイト0だけを設定して表示を行っているぶんには，モード03Hと見た目は同じで，疑似ビデオバッファの機能がない分やや不便なだけです．

パレット

パレットレジスタの設定（AH＝10H INT 10H）の機能は，一見グラフィックモードだけの機能のように思えますが，実際はテキストモードでも有

メモリ用語集
増設メモリ　1MBを超える空間にマッピングできるメモリのこと．プロテクトメモリとも呼ばれ，プロテクトモードや仮想86モードからリニアにアクセスできる．

効です．つまり，前景・背景各16色指定可能な“色属性”は，特定の色に関係づけられているのではなく，単に16色のパレットにCGA互換のデフォルトの配色が設定されているにすぎません．

このことを利用すると，たとえばDOSのデフォルトの画面色(07H)を，アトリビュートはそのままで，「青い背景に明るい白」などに変更することができます（リスト5）．

グラフィックプログラミング

モード12Hをはじめとするグラフィックモードでは，実用上，日英のモードの違いをあまり意識する必要はないでしょう．事実，英語モード用のプログラムでも，グラフィックモードを使用するものは多くの場合「日本語モードのグラフィック画面」上でそのまま動作します．たとえば英語モード用のWindows3.0などは日英どちらのモードからでも起動できます．

プログラミングツールでいえば，たとえばQuickBASICは，それ自体の統合環境はテキストモード上で動作するため日本語モードからでは動きませんが，コンパイルずみのプログラムの実行時には，グラフィックのモードであれば日本語モードでもそのまま動作します．PRINT命令で日本語の表示さえ可能です．ただしカーソルがカタカナの「ロ」に化けて表示されるのがご愛敬ですが……．

スクロール

グラフィック画面を用いてテキスト画面をエミュレートする関係上，DOS/Vマシンはこれまでの国産パーソナルコンピュータより，宿命的に「フォントの表示が遅い」，「スクロールが遅い」などの問題が生じやすい特質をもっています．もっとも，実際に使ってみて「案外速い」と思われた方もいらっしゃるかもしれません．

DOS/Vのスクロールが「思ったほどは遅くない」理由は，1つには，VGAがもっている「ハードウェアスクロール」の機能を使用して，スクロールを高速化する工夫が施されている点にあります．

このハードウェアスクロールというのは，要するに全画面を一挙に上下にスクロールさせる機能で，これを使う場合にはそこそこのスピードが確保されるのですが，たとえば，

「画面の最下行はそのままで，上の24行分だけスクロールさせる」

ということは本来できません．そこでこのような場合，$DISP.SYSは，

1．全画面をスクロールさせる
2．本来スクロールさせるべきでなかった行を書きなおす

という処理を自動的に行うようになっています．このため，どうしても画面の上下の行に，独特の「ちらつき」が生じます．

一方，1行目から12行目にかけてスクロールさせるとか，画面の左半分だけをスクロールさせるとか，左右に罫線キャラクタで枠を表示して内側だけスクロールさせるとかいう場合，上のような方法はとれませんから，ちらつきは出ないが速度が遅い「ソフトウェアスクロール」の手法がとられます．

ハードウェアスクロールが使われるか使われないかは状況によって$DISP.SYSが自動的に判断して切り替えるわけですから，プログラマが意識する必要は基本的にはないのですが，「なるべく高速にスクロールさせたい」，逆に「ちらつきの出ないようにしたい」という場合，スクロールするウィンドウの範囲を調整することによって，プログラマ側がある程度意識的にハードウェアスクロールの有無をコントロールすることができる，という点は意識しておいたほうがよいかもしれません．

“DOS汎用ソフト”とDOS/V

いわゆる「MS-DOS汎用」のソフトウェアは，ほんとうにそれがMS-DOSの公開機能だけを用いているな

リスト5

```
procedure SetPalette(Palette,ColorNumber:byte);
begin
    with Regs do begin
      AL:=$00;              {個別指定}
      BL:=Palette;          {設定するパレット・レジスタの番号0..15}
      BH:=ColorNumber;      {カラー・レジスタ番号 0..63}
      Intr($10,Regs);
    end;
end;
begin
   SetPalette(0,8);         {dosの背景色をダークブルーに}
   SetPalette(7,54);        {dosの文字の色を明るい黄に}
end;
```


メモリ用語集
タイムシェアリング　マルチタスクシステムにおいて，タイマを利用して一定時間ごとにタスクを切り替えること．

ら当然DOS/Vでも動作します．たまにパソコン通信で「DOS/V用のLHAやISHはどこにありますか？」といった質問を見かけることがありますが，これらは当然そのまま動きます．

ただし，ANSIエスケープシーケンスを用いてカーソル位置，表示色などの画面制御を行っているソフトは，DOS/Vでは「うまく動作しない場合がある」ことを頭に入れておく必要があるでしょう．

理由は2つあります．1つは，もともとIBM DOSではANSIエスケープシーケンス対応機能がオプション扱いだからで，デバイスドライバANSI.SYSを組み込まないと利用できないのです．

もう1つの理由は，ANSI.SYSの機能自体，多くの日本語パソコンと完全互換でない点です．たとえばカーソルやファンクションキーの表示／非表示，スクロールなどの機能はDOS/Vも含めたIBM DOSのANSI.SYSではサポートされないため，これらのコードを出力するプログラムを実行すると画面が乱れてしまいます．なお，使用可能なエスケープシーケンスについては『IBM DOSバージョン5.0/Vユーザーズ・ガイド』の「ANSI.SYS」の項に詳しく説明されています．

DOS/V キーボード

キーボードの入出力に関しては，日英のキーボードの見た目の差ほどプログラマ側がモードの差を意識する必要はありません．

日本語モード用に追加されている「変換」，「無変換」，「半角／全角」などのキーは，あってもなくてもどのみち日本語入力FEP以外のプログラムが通常は直接アクセスしないキーですし，それらのキーが存在しないキーボードをユーザーが使っている可能性もあるためです．

また，PCでは，DOS，BIOS割り込み，どちらの方法を用いてもDBCSのコードを問題なく取得できます．つまり，漢字の1バイトめにあたるコードがキーボードから入力されたら次に入力されたコードを待って，1つの全角文字として取り扱ってやればそれで終りです．

$IAS.SYS

DOS/Vでは，標準的な日本語入力用インターフェースとして，この$IAS.SYSというデバイスドライバがデフォルトで組み込まれることになっています．

この$IAS.SYS自体は，いわゆる日本語入力FEP(かな漢字変換プログラム)ではなく，それらとキーボードとの間に入って，ローマ字かな変換，ひらがな，カタカナの入力を提供したり，かな漢字変換モードのオン／オフを制御したりしています．また，Windowsや OS/2などのマルチタスク環境での日本語入力環境の確保の役割も担って

表5　$IAS.SYSによって拡張されるINT16Hキーボードサービス(日本語モード)

$IAS.SYSがインストールされていないときは一般に結果は不変です．

AHの値	機能内容	呼び出し時の設定	レジスタに返される値	備考
13H	2バイト文字セットの状況モードの設定読み取り	AL＝00H　設定 DL＝右欄参照 AL＝01H　読み取り	DLのビット 7　0＝非漢字　1＝漢字 6　0＝JIS　1＝ローマ字 5〜3：予約ずみ 2〜1：00英数 01カタカナ 10ひらがな 11予約ずみ 0　0＝半角　1＝全角	
14H	シフト状況の表示・消去	AL＝00H　表示 AL＝01H　消去 AL＝02H　状況読み取り	AL＝00H　表示中 AL＝01H　消去中	

メモリ用語集
デアロケート（deallocate）　アプリケーションに与えた論理ページを解放し，ほかのアプリケーションからも使えるようにすること．

います.

実際に発売されているDOS/V対応の日本語入力FEPのすべてがこの$IAS.SYSに対応しているわけではないのが少々不安ですが, $IAS.SYSに対応している日本語入力FEPの場合には, 共通のやり方(INT 16Hのサービス群) でかな漢字変換モードのオン／オフ, 状況の取得, かな漢字変換用のシステム行のオン／オフなどができるようになっています(表5).

キーボードタイプ速度の設定

これも日英どちらのモードにも共通の機能なのですが, 国産のパソコンに比べてPCのデフォルトのキーボードのコード発生間隔(リピートレート)は遅く, これが原因で「DOS/Vは486マシンでさえスクロールが遅い」みたいに思い込んでいる人がいたりします.

つまり, 実際にはこの「キーボードタイプ速度の設定」の機能で, リピートレートを最速にしてやるだけで大幅な「スクロール速度の改善」ができてしまったりするわけです. ただ, この設定はDOS標準のMODEコマンドでユーザーが変更可能ですので, 個々のプログラムがこの機能を使ってタイプ速度を勝手に変更してしまうのはあまり好ましくないかもしれません.

フリーソフトが開いたDOS/Vの新しい魅力

DOS/Vの日本語環境を語るとき, 標準機能ではないけれどもいまや大きな魅力に数えられる2つのデバイスドライバがあります.

1つは, Windowsばりの「マルチフォント環境」をDOS上で実現する$FONTX.SYS.

もう1つは, テキスト画面の表示文字数や行数を, 画面の解像度にあわせて大幅に拡張できる「ハイテキストドライバ」$DISPH.SYSです.

どちらも日経MIX(ibm.pc会議)やNIFTY-Serve(IBMフォーラム)など, パソコン通信上のプログラムライブラリで入手できるフリーソフトウェアとして公開されているもので, ファイル名が物語るとおり, 前者はDOS/V標準の$FONT.SYS, 後者は$DISP.SYSと置き換えてDOS/Vの機能を拡張します.

両方あわせると, たとえば次のようなことが簡単にできてしまうのです.

・16ドット100桁×37行の広大なテキスト画面に, フォントは小さめでも見やすいボールド体と「丸文字」をコマンド1つで切り替えたりできる. フォントファイルさえ用意すれば, PC98固有の記号文字なども表示可能.

もしかしたら, これらは標準のDOS/Vの将来の姿を予言するものかもしれないと思わせる面もありますので, これら2つのソフトを利用してできること, 対応ソフトを開発する上での注意点をあげておくことにします.

$FONTX.SYSとマルチフォント環境

$FONTX.SYSは, 次のような点で純正の$FONT.SYSの制約を解消できる仕様をもったデバイスドライバです.

1. フォントの記憶領域を, BIOSのINT15Hインターフェースによらず, XMSメモリやEMSメモリ, 必要ならばメインメモリにも置ける.
2. 文字の縦横ドット数を任意に設定可能.
3. フォントファイルの構造が公開されており, また, そのフォーマットもシンプル.
4. フォントパターンのすべてがいつでも再定義可能.
5. シフトJISで表現可能なすべてのコードが使用可能.
6. (メモリに余裕があれば)複数のフォントパターンをロードし, 任意に切り替え可能.

とくに筆者が重宝しているのはフォ

表6

モード番号	機　能	解像度
8BH	80x30テキストモード	(640×480)
8CH	80x30テキストモード（73h 互換）	(640×480)
82H	グラフィックモード12Hの800x600版	(800×600)
83H	100x31テキストモード	(800×600)
84H	100x31テキストモード（73h 互換）	(800×600)
85H	100x37テキストモード	(800×600)
86H	100x37テキストモード（73h 互換）	(800×600)

メモリ用語集
タイムシェアリング　マルチタスクシステムにおいて, タイマを利用して一定時間ごとにタスクを切り換えること.

リスト6

```
program SpaceToTofu;
uses Dos;
procedure AccessFontPat(Read_or_Write:byte;
                         BuffSeg,BuffOfs, FontSize, SJISCode  :word;
                       var ResultCode :byte);
var Regs:Registers;
begin   with Regs do begin
          AH:=$18;               {機能番号 18h}
          AL:=Read_or_Write;     {書き込みなら1、読み込みなら0を指定}
          BX:=$0000;             {0でなければならない}
          CX:=SJISCode;          {シフトJIS コード。半角のとき CH=0}
          DX:=FontSize;          {DH:=幅 DL:=高さ。例:8x16なら$0810}
          ES:=BuffSeg;           {パターン読み書き用に用意したバッファの}
          SI:=BuffOfs;           {セグメントとオフセット}
          Intr($10,Regs);
          ResultCode:=AL;        {エラーなら0以外が返る}
        end;
end;
const FontWrite=1;
      FontRead=0;
var Zen16pat:array[0..Pred((16 div 2)*2)] of byte;
    Result:byte;
begin {main}
   FillChar(Zen16pat,SizeOf(Zen16pat),#$FF);
   AccessFontPat(FontWrite,Seg(Zen16pat),Ofs(Zen16pat),$1010,$8040,Result);
end.
```

ント再定義機能で，現在表示中のフォントを太らせて「ボールド」表示にしたり，90度回転させて「縦書きDOS環境」(55noteにキーボードを外付けし，55note本体は屏風のように立てて使うと快適）を作ってみたり，いろいろ遊んで？ いる今日このごろ．

DOS/V用日本語MS-Windowsの全角システムフォント(10ポイントの)はDOSのフォントそのものなので，「丸文字システムフォントのWindows」などというものも可能になります．

なお，フォントの読み出し・再定義方法はもともとのBIOSの機能そのままで，オリジナルでは外字領域以外に書き込みができない（エラーコードが返る)ところがエラーにならないだけです．たとえばリスト6のプログラムを実行すると，シフトJISコード8040Hの全角空白が■(トーフ)に変身します．

$DISPH.SYSとハイテキスト環境

DOS/Vオリジナルの$DISP.SYSとの差分のかたちで提供されており，800×600ドット16色のグラフィックモードをもつSVGAのシステムにこれを組み込むと，現在のDOS5.0/V対応版の場合には表6のモードが拡張されます．

8BH, 8CHのモードは通常のVGAの解像度の範囲内ですので，液晶ディスプレイのノートパソコンなどでも利用できます．

画面の桁数・行数が増える以外はモード03hまたは73hのテキストモードの機能と同じですから，桁数・行数を80×25と“決め打ち”せず，あらかじめ可変の要素と考えてプログラミングしさえすれば対応は簡単です．

この画面の桁数・行数は，次に示すアドレス(BIOSのワークエリア)に格納されていますので，簡単にチェックすることができます．

0040:0084h　表示可能行数−1
0040:004Ah　表示可能な1行の文字数

英語モードではもともと43行(EGA)，50行(VGA)テキストモードがサポートされていますが，これらに対応する際にもこの情報が役にたちます．

なお，03H互換テキストモードで疑似ビデオバッファにアクセスする場合，画面の表示文字数によって行とアドレスの関係が変化することになりますので注意が必要です．

すでに市販のエディタ「Vz」がバージョン1.57でこのドライバに対応しており，「ハイテキスト環境」も考慮した日本語FEPも登場してきました(WX II＋)．いずれこうしたハイテキスト環境は，DOS/Vでは常識化することが予想されます．

これからのDOS/V

誕生当初，「日本アイ・ビー・エム社のノートパソコン用DOS」という程度の受け止め方しかされていなかった面もあるDOS/Vですが，可能性としては，DOS/Vはこれまでのパソコンのユーザー層とは異なった部分で支持を集めることのできるDOSなのではないか，と筆者は思っています．

その“可能性”の中から，すでに具体化して我々の目の前にある部分を原稿の中にもできるだけ反映させたつもりなのですが，いかがでしたでしょうか．64色中16色，フォントは自由に書き換え可能．100×37行の広大なテキスト画面……これからさらに何が出てくるかはわかりませんが，とにかくいまあげた環境なんかは，プログラミングの経験のある方なら，かなり捨てがたい魅力に感じられるのではないかと思うのですが，さて……？

メモリ用語集
DPMI　1MBを超えるメモリ空間を複数のアプリケーションから利用するためにMicrosoftをはじめとする米国大手ソフトベンダ11社で提唱された規格．

特集
新しいDOS環境を
探検する

DOS/V

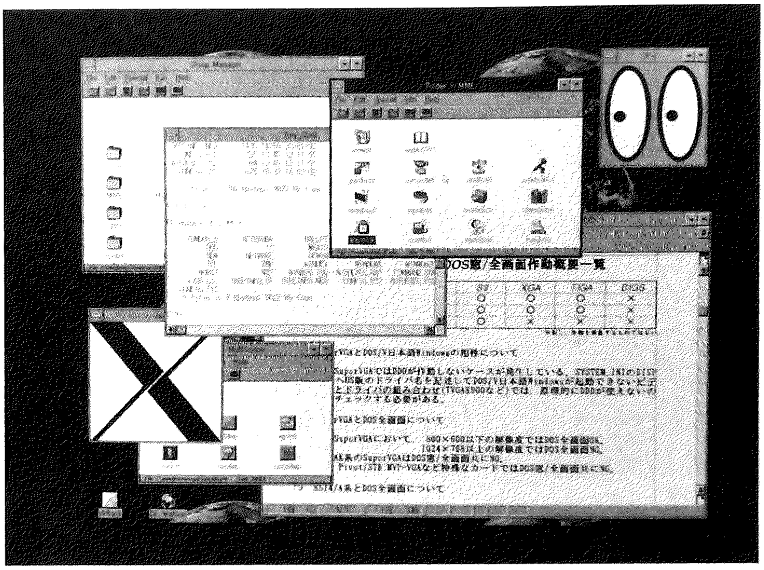
DOS/Vパソコンの開発環境

プログラマの理想郷を探る

田中亘

DOS/Vを取り巻く環境が，にわかに活気づいてきている．いままでは，日本で売れるハードはPC-9801，ソフトもPC-9801用のものと相場が決まっていた．しかし，昨年の暮れごろから情報に敏感なハイエンドユーザーやプログラマを中心に，DOS/Vマシンの売れ行きが急激に伸びてきている．今回は，DOS/Vマシンを用いてプログラムの開発を行っているC・F・Computingの西川氏を訪ねて，氏の開発環境やプログラマにとってのDOS/Vのメリットについて話をうかがった．

驚きの大画面



▲写真1　X-Window?

まず，写真1の画面を見てもらいたい．一見するとUNIXのX-Windowのような画面に見えるが，これはX-Windowなどではなく，日本語MS-Windows 3.0の画面だ．ナナオ製FLEXSCAN T560;, 17インチモニタ上で，1280×1024ドットの高解像度表示，256色同時発色という環境で日本語MS-Windowsを表示したものだ．当然のことだが，日本語を使用することも，日本語MS-Windows 3.0対応の日本語アプリケーションを利用することも可能である．

ここで，西川氏が利用しているハードウェア構成を紹介しておこう．使用しているパソコンは，見てのとおり(写真3)，アメリカなどで売られているごく普通のIBM PC/AT互換機である．CPUは33MHzの80486, 200MBのハードディスクと8MBのメモリを実装している．ビデオボードにEIZO AA51(8514/A互換)高解像度ボードを利用し，さらにマルチメディアに対応させるためにサウンドブラスタープロとCD-ROMドライブを搭載している．また，オフィスにある複数のIBM PC/AT互換機は，すべてNetWareによるLANによって接続されている．

DOS/V用の開発言語とエディタ

プログラマが最初に必要とするツールは，なんといってもコンパイラとエディタだ．

現在のところ，DOS/V用として正式に対応している日本語版コンパイラは，Microsoft C Professional Development System (以下MS-C)および日本アイ・ビー・エムから発売されている言語製品のみである．こう聞くと驚かれる方も多いと思う．これでは，どうしようもないではないかという人もいるだろう．しかし，ここでDOS/Vが日英両用のバイリンガルOSだということを思い出して欲しい．英語モードでは，アメリカでIBM PC/AT用として発売されているすべての処理系が使用できるのである．コンパイラやリンカ，デバッガなどの開発ツールの大半は，日本語の処理を必要としない．このことを理解すれば，開発ツールの選択に迷うことはあっても，開発ツールがなくて困ることはない．

英語版のコンパイラを利用したときに問題になる部分は，日本語2バイト文字の表示くらいだろう．この問題は，ソースコードの内部には直接日本語を入力しないで，日本語データだけを別途用意することで解決できる．開発の

▲写真2　西川和久氏
◀写真3　西川氏が開発に使用しているものと同型のIBM PC/AT互換機

ときはすべて英語にしておき，最終的に日本語のメッセージテーブルデータだけを用意して，メッセージを切り替えられるようにすれば，マルチリンガル対応のプログラム開発も簡単に実現できる．メッセージへのアクセスは，ポインタなどを使えば容易だし，プリプロセッサなどで日本語コードを変更する必要もない．

DOS/V対応の日本語版コンパイラは少ないが，英語だけならば最新のコンパイラを利用できるという点で，DOS/Vのマルチリンガル環境のほうが有利だといえる．

また，最新のコンパイラが利用できるのと同様に，最新のデバッグ環境を利用することも可能である．コンピュータユーザー人口もプログラマ人口も日本とは比べものにならないほど多いアメリカには，ハードウェアと併用することでMS-DOSのカーネルまでデバッグできるといわれるものや，ICE（IN-CIRCUIT EMULATOR）と同等の機能をソフトウェアだけで実現するなどという優れたデバッガが数多く販売されている．

英語だけしか使えなくても，アルゴリズムの構築や，ロジックのデバッグならば，何の問題もない．英語上ですべてのルーチンやアルゴリズムをチェックした後で，最終的なメッセージや2バイト文字処理だけをチェックすれば，すぐにDOS/V対応の日本語アプリケーションが完成するのだ．

このように，PC-9801などに移植されない高度な開発ツールや言語処理系の最新バージョンをつねに使用できるというのが，プログラマがDOS/Vを利用するもっとも大きな魅力といえるだろう．

また，今年の後半までには，日本語モードに対応した処理系が国内各社から発売されると予測されている．

次にエディタだが，これは，PC-9801でも人気の高いMIFESとVz，さらにプログラミング用に最適化されたエディタとしてアメリカでプログラマから熱い支持を受けているBriefがDOS/V用として発売されているので，なにも問題はない．

COLUMN　高機能ビデオボード

IBM PCアーキテクチャのマシンでは，ビデオボードを交換するだけで簡単にグラフィック機能を高めることができる．

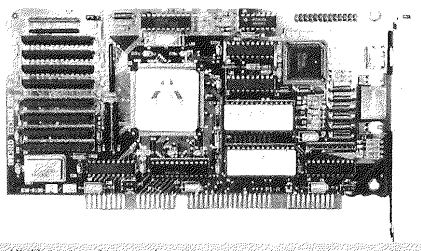
Windowsのようにグラフィックに大きな負荷がかかるソフトウェアを動作させる際には，このビデオボードの性能が大きく関わってくるのである．現実に，普通のVGAのビデオボードを備えた486マシンよりもWindows用に最適化されたグラフィックアクセラレータを積んだビデオボードを備えた386SXマシンのほうが，高速にWindowsが動作する．

また，スピードだけではなく解像度も，800×600，1024×768，1024×960，1280×1024など，さまざまに機能アップすることが可能だ．表示できる色数にしても，16色からフルカラーまで，各種のボードが発売されている．

これらのボードの中で，コストパフォーマンスの高さでWindowsユーザーの注目を集めているのが，S3（エススリーではなくエスキューブドと発音する）というプロセッサを搭載したものだ．通常のVGAの約10倍のスピードでWindowsが動作するようになるというのだから，その効果は絶大である．

以下にS3プロセッサを搭載した代表的なビデオボードをあげておく．

Actix Quantum　399ドル
CSS MaxGraphics 495ドル
Diamond Stealth VRAM　349ドル
Diamond Stealth HiColor　445ドル
Orchid Fahrenheit 1280　499ドル
STB Wind/X　499ドル



S3プロセッサを搭載したビデオボード

Windowsアプリケーションの開発環境

一方，DOSから日本語MS-Windows 3.0に目を向けてみると，日本アイ・ビー・エムから発売されているMS-C+SDK（Software Development Kit）といったオーソドックスな開発ツールに加えて，QuickC for WindowsやTurbo C++ for Windowsなどのまだ日本語化されていない処理系を利用することができる．Windowsが完全な互換性をもっていれば，PC-9801のWindows上でもこ

メモリ用語集
デバッグレジスタ　80386以降のCPUでサポートされたブレークポイントを指定するデバッグ用のレジスタ．

れらの処理系を利用できるはずなのだが，なぜかいろいろと問題が生じることが多いようである．いうまでもないが，IBM PC/AT互換機＋DOS/V＋日本語Windowsにおいてはまず問題が生じるようなことはない．

また，PC-9801のWindows用として日本語化された処理系やツールを，DOS/V用Windowsで使用することも可能である．たとえば，Borland C＋＋に入っているリソースワークショップを抜き出して，DOS/V上の日本語Windowsで利用できる．PC-9801用のツールをDOS/Vで利用するときに注意する点は，メディアの変更だけだ．2DDの9セクタフォーマットを経由すれば，PC-9801からDOS/Vへのデータ転送は問題なく行える．

ここで，ちょっとした裏ワザを披露しよう．英語版Borland C＋＋や英語版Turbo Pascal for Windowsに付属しているTurbo Debugger for Windows（TDW）は，普通は日本語Windowsから利用できない．なぜならば，TDWはDOS上で動作するデバッガで，DOS/Vの日本語モードでは動作しないからである．しかし，英語モードならば正常に動作する．そこで，日本語WindowsのWIN.COMを英語版WindowsのWIN.COMと交換してやる．こうすると，DOS/Vの英語モードで日本語Windowsが起動するようになり，日本語Windows上から英語版TDWが利用可能になる．お試しあれ！

また，Multiscope Debuggers for Windowsという，ソースコードやレジスタウィンドウを開いて，統合的にWindows上でデバッグできる製品もアメリカでは発売されているので，これらを利用してもいいだろう．

COLUMN　IBM PC用高機能デバッガ

● Soft-ICE/W

Soft-ICE/Wは，DOS用のデバッガとして人気のあったNu-Mega Technologies Inc.のSoft-ICEのWindows対応版である．Soft-ICEは，その名の通りハードウェアのデバッギングツールであるICE（IN-CIRCUIT EMULATOR）を使用しなければ実現できなかったハードウェアブレーク機能を80386マシン上でソフトウェア的に実現したデバッガである．ほかにも，メモリ常駐プログラムやデバイスドライバのソースレベルデバッグ，メモリ常駐プログラムとアプリケーションプログラムなど2つのプログラムを同時にデバッグする，といった機能をもっている．また，Soft-ICEは拡張メモリ上で動作するので，メモリを大量に消費するプログラムのデバッグなども問題なく行える．

Nu-Mega Technologies Inc.は，Soft-ICE/W以外にも，Cプログラムによくあるポインタの初期化し忘れなどによる範囲外のメモリアクセスをトラップするBOUNDS-CHECKERや，CodeView for Windowsを1つのディスプレイで動作させるCV/1などのツールを発売している．

Soft-ICE/W　386ドル
BOUNDS-CHECKER　249ドル
CV/1　129ドル
Nu-Mega Technologies Inc.
P.O.Box 7780
Nashua, NH 03060-7780 U.S.A.
CALL(603)889-2386
FAX(603)889-1135

● PERISCOPE DEBUGGER

PERISCOPE DEBUGGERには，ソフトウェアのみのものからIBM PC/AT互換機のスロットに差して使用するボードを含むものまで，複数のモデルが存在している．最上位のモデルでは，どのようなときでも自由自在にハードウェアブレークをかけられるので，MS-DOSのカーネルまでデバッグできるといわれるほどである．

実際，アメリカのAddison Wesleyから発売されているUNDOCUMENTED DOSの著者達がDOSの内部探検に使用したのが，このPERISCOPE DEBUGGERだといわれている．

PERISCOPE DEBUGGER　195ドル〜
The Periscope Company, Inc.
1197 PEACHTREE ST.
PLAZA LEVEL
ATLANTA, GA 30361
CALL(404)875-8080
FAX(404)872-1973

● MULTISCOPE DEBUGGERS

Windowsプログラムのデバッグを行うには，通常，SDKに付属しているCodeView for Windowsか，またはBorland C＋＋，Turbo Pascal for Windowsに付属しているTurbo Debugger for Windowsを使用する．しかし，これらのデバッガはDOS上で動作するもので，Windowsプログラムではなかった．

MULTISCOPE DEBUGGERSは，これらのデバッガとは異なる完全にWindows上で動作するいまのところ唯一のデバッガである．MULTISCOPE DEBUGGERSがデバッグできるのは，CodeView互換のデバッグ情報を含むオブジェクトとなっている．Windowsプログラムだけではなく，DOSプログラムのデバッグも可能である．

MULTISCOPE DEBUGGERS　379ドル
MultiScope, Inc.
1235 Pear Avenue
Mountain View, CA 94043
CALL(415)968-4892

メモリ用語集
動的システム　ソフトウェアのサイズが実装メモリよりも大きく常駐部分と非常駐部分に分けて実行するもの．

グラフィックアクセラレータボードとDDD

実際，Windowsアプリケーション開発マシンとして検討したときに，DOS/Vパソコンほど最適なマシンはないのではないだろうか．ハイパワーなマシンを安価に入手できるというコストパフォーマンスの高さもさることながら，Windowsを快適なスピードで動かすことができるのは大きな魅力である．

IBM PCでは，国産マシンとは異なり，ビデオシステムがビデオサブシステムとして完全にマザーボードから独立している．ビデオサブシステムは拡張スロットに差し込む1枚のボードであり，ユーザーは必要に応じてより多くの色が表示できる，またはより高精細なビデオボードを購入，交換することでシステムのパフォーマンスを向上させることが可能になる．MS-Windows 3.0が発売されて以来，アメリカではWindowsを高速化するグラフィックアクセラレータつきのビデオボードが次々に開発されている．これらのビデオボードを利用することで，Windowsを標準のVGAの10倍～30倍も高速化できるのだ．S3社の86C911チップセットを搭載したビデオボードも，そのようなグラフィックアクセラレータつきビデオボードの1種である．

快適なビデオボードとWindowsの環境下でソフト開発ができれば，Windows対応アプリケーションの開発期間も，かなり短縮できるはずだ．

しかし，ここで1つ問題が生じる．新しいビデオボードでWindowsを使用するためにはそのボードに合ったWindows用ディスプレイドライバが必要になる．アメリカで開発されたビデオボードには，英語版Windows用のディスプレイドライバは付属していても，日本語版Windows用のディスプレイドライバは付属していない．これではせっかく高速なビデオボードを入手しても宝の持ち腐れである．そこで，C･F･Cが開発したのがDDD（Windows Display Dispatch Driver）だ．DDDとは，英語版Windowsディスプレイドライバをそのまま使い，WIFEに対応する必要があるAPIだけをディスパッチするドライバである．これによって，ビデオボードごとに日本語ディスプレイドライバを開発する必要がなくなり，海外で開発されたビデオボードをタイムラグなしに利用できるようになった．

開発環境としてのWindows

ここで，実際にプログラミングを行うときのデスクトップを見てみよう．Windowsがデフォルトで使用されているのはいうまでもないが，さらにWindowsの使い勝手を上げるためにSloop Managerが使用されている．

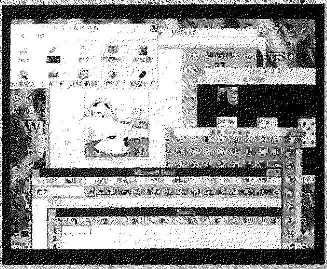
ソースを作成するためのエディタとしては，アメリカ製のシェアウェアで

COLUMN　DDD(Display Dispatch Driver)

海外で発売されている高機能ビデオボードをDOS/V＋日本語Windowsで使用する際に問題となるのが，日本語Windows用ディスプレイデバイスドライバである．海外で発売されているビデオボードには，そのビデオボードを開発したメーカーが作成したWindows用ディスプレイデバイスドライバが付属してくるが，これはWIFEに対応していないため日本語Windowsでは使用できない．そこで，CFC（C･F･Computing）が開発したのが，DDD（Display Dispatch Driver）である．

DDDは，英語Windows用ディスプレイデバイスドライバをWIFEマネージャに対応させるためのディスパッチドライバだ．DDDによって，いちいちビデオボードに合わせて日本語Windows用ディスプレイデバイスドライバを開発する必要がなくなったわけである．おかげで，海外で発売されているグラフィックアクセラレータを搭載したビデオボードなども，なんのタイムラグもなしに日本語Windowsで利用できる．

DDDは，秋葉原のT-ZONEをはじめとするIBM PC互換機取扱店にて購入することができる．



46

メモリ用語集
DOSエクステンダ　MS-DOSから1MBを超えるメモリ空間を利用可能にする一種のドライバ．ライブラリとしてコンパイル時にアプリケーションに組み込まれるものもある．

あるWinEditを使用している．これは，コンパイラのエラーメッセージをもとにソースコードを自動的に呼び出すといった統合環境のような開発環境を実現する優れものである．残念ながら日本語の使用については，多少難がある．

通常はこのWinEditでプログラムを開発し，日本語表示用のデータが必要になったときだけ，MS-Word for Windowsの日本語版でデータを入力する．もし，PC-9801の日本語MS-Windows 3.0でMS-Wordを利用した経験のある人ならば，この環境に疑問を抱くかもしれないが，先ほど述べたように高速なビデオボードを用いることで，DOS/Vマシンでは十分快適にMS-Wordを使用することができるのである．それも，PC-9801とは比べものにならない広い画面で．

DOS/V＋日本語Windowsという環境のメリットに，DOSの日本語モードで動作するプログラム，DOSの英語モードで動作するプログラム，Windowsで動作するプログラムを同時に実行できるという点がある．この利点を十分に生かし，1つのDOSウィンドウで「UNDOCUMENTED DOS」という書籍に付属してくる割り込みリスト表示プログラムを起動し，もう1つのDOSウィンドウでMicrosoft Programmer's LibraryというCD-ROMソフトの検索プログラムを起動して，いつでも参照できるようにしている．これらの技術資料をコンピュータの前を離れることなく迅速に検索・参照できるため，机上にリファレンスマニュアルを置く必要がなくなったという．とくにMicrosoftのCD-ROMにはDOSのファンクションからWindowsのAPIまで，プログラミングに必要なあらゆる資料がデータベー

COLUMN　IBM PCプログラミング日本語文献

●THE IBM PC & PS/2プログラマーズガイド

「THE IBM PC & PS/2プログラマーズガイド」は，ベストセラーソフトウェアであるノートンユーティリティーズの作者として知られるPeter Norton氏が書き下ろしたIBM PCプログラミングの入門書である．

内容は，IBM PCおよびPS/2のハードウェアの構成から，ROM BIOSの解説，MS-DOSの基礎知識，さらにこれらを利用したプログラミングの技法まで，IBM PC系のマシンに関する全般を理解できるように構成されている．

IBM PC，PS/2およびその互換機を用いてプログラミングを行う際に，最初に読むべき本といえるだろう．

THE IBM PC & PS/2プログラマーズガイド
Peter Norton/Richard Wilton著
SE編集部訳・編
5200円
翔泳社

●PC & PS/2ビデオシステムプログラマーズガイド

IBM PCおよびPS/2では，グラフィック関係のシステムがビデオサブシステムとして独立している．「PC & PS/2ビデオシステムプログラマーズガイド」は，このビデオサブシステムを利用したプログラミングの解説書である．

CRTの基本原理からビデオBIOSの利用法まで，さまざまなプログラミングテクニックが紹介されている．取り上げられているビデオボードは，MDA，CGA，EGA，HGC，HGC+，MCGA，VGAである．

PC & PS/2ビデオシステムプログラマーズガイド
Richard Wilton著
SE編集部訳・編
6000円
翔泳社

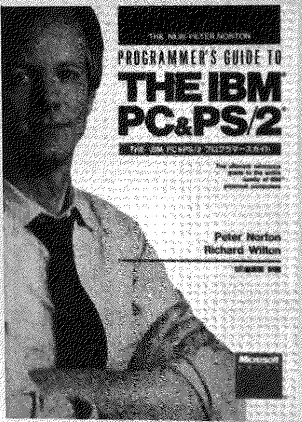
●プログラマーのためのPCソースブック

この「プログラマーのためのPCソースブック」は，IBM社のPCに関するあらゆるテクニカル情報を1冊に凝縮させたリファレンスブックである．

この本は，以下に示すリファレンスマニュアルに掲載されている情報をカバーしている．

- IBM personal computer Technical Reference(PC，XT，AT，PS/2)
- IBM BIOS Technical Reference
- IBM DOS Technical Reference
- MS-DOS Technical Reference
- Microsoft Windows Development Kit
- Microsoft Mouse Reference
- Lotus/Intel/Microsoft EMS Specifications
- Intel Microprocessor Handbooks

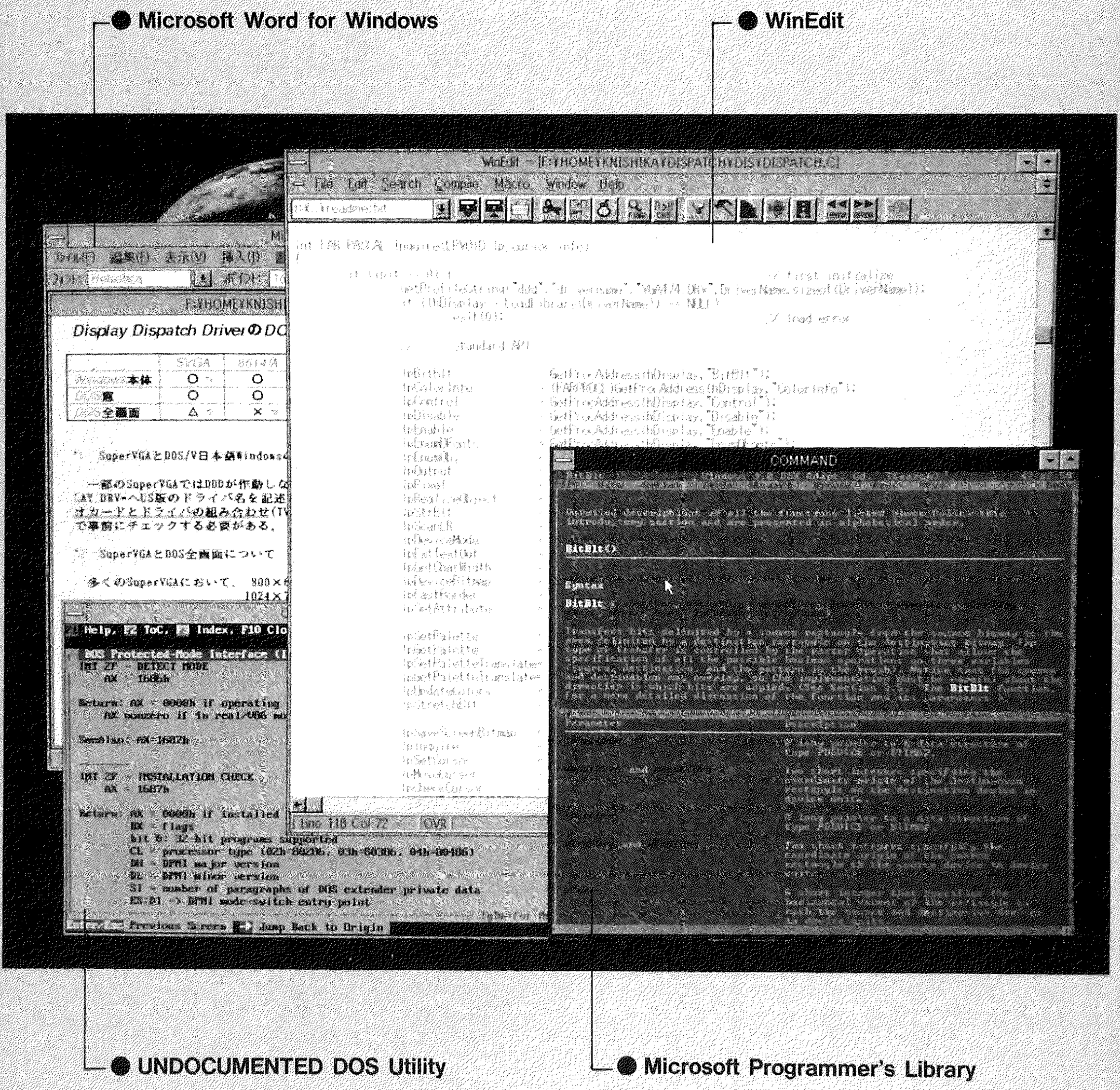
プログラマーのためのPCソースブック
Thom Hogan著
SE編集部訳・編
5200円
翔泳社



メモリ用語集
BIOS (Basic Input Output System)　基本入出力システムと訳し，ディスク，CRTなどの基本的な入出力サブルーチンの集まり．システムコールを使って利用する．

COLUMN　DOS/V+日本語Windowsによるプログラミング環境

ここに示すのが，プログラミングを行う際のデスクトップである．ナナオ FLEXSCAN T560i，17インチモニタに表示された1280×1024，256色の日本語Windows上で，さまざまな開発ツールを用い，オンラインマニュアルを参照しながら快適にプログラミングを進めることが可能である．



メモリ用語集
バインダ　動的システムの開発でオブジェクトファイルを結合するユーティリティ．

● **Microsoft Word for Windows**

日本語ワードプロセッサであるMS-Wordを用いて，日本語ドキュメントおよびプログラムで使用する日本語のコメントなどを作成する．MS-WordはWindowsアプリケーションであるため，簡単に図版や表などを作成して文章中に取り込むことができる．

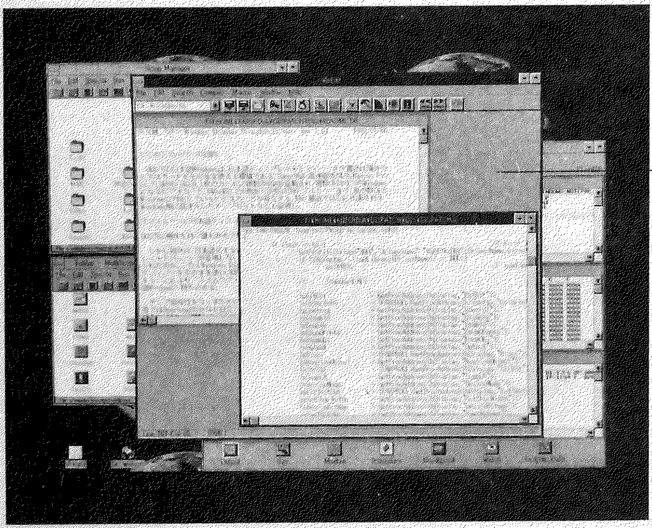
● **UNDOCUMENTED DOS Utility**

Addison Wesleyから出版されている「UNDOCUMENTED DOS」に付属しているディスクに含まれているユーティリティプログラム．未公開（アンドキュメント）情報も含めて，MS-DOSに関するあらゆる情報がハイパーテキストデータベースとしてまとめられている．

このユーティリティは，DOSウィンドウの1つを英語モードにすれば利用できる．このようなことが簡単に実現できるのが，DOS/Vの強みだ．

● **Microsoft Programmer's Library**

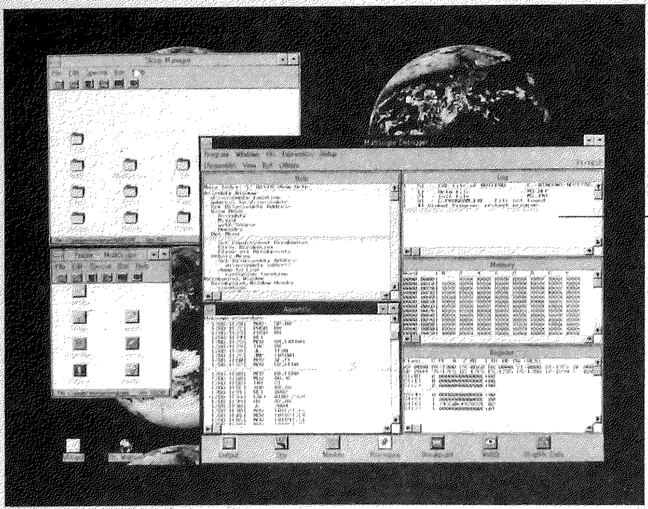
米国Microsoft社が発売しているプログラマ向けCD-ROMソフトウェアであるMicrosoft Programmer's Libraryを，やはりDOSウィンドウを英語モードにして活用している．数十メガバイトを越える膨大なプログラミング情報・サンプルソースを即座に検索・参照することができるため，机の上にリファレンスマニュアルなどを置いておく必要がなくなる．



● **WinEdit**

アメリカで開発されたWindows対応のプログラミングエディタ．ソース作成からコンパイル，デバッグまでを統合して行える．マイクロソフト社のPWB（Programmer's WorkBench）やボーランド社のIDE（Integrated Development Environment）のような環境を作り出すことができる．また，日本語の入力はできないが表示することは可能なので，日本語ドキュメントの参照にも利用できる．

WinEditは，シェアウェアとしてパソコン通信などを通じて配布されている．



● **MULTISCOPE DEBUGGERS**

Windows上で動作するデバッガであるMULTISCOPE DEBUGGERSを利用している．CodeView for Windowsは2つのモニタを必要とするが，MULTISCOPEを利用すれば1つのモニタでWindowsプログラムのデバッグが可能になる．RS-232CだけでなくNetBIOSを利用したリモートデバッグなど，CodeViewにない機能を備えた高機能ソースレベルデバッガだ．

メモリ用語集

バンクメモリ　メインメモリ80000Hから9FFFFHまでの128KBをメモリページとし，ポートをアクセスすることによって切り替えるメモリ拡張方式

スとして収められており，プログラミングの効率アップにこれ以上のものはないそうだ．

また，デバッグの際には，2つのディスプレイを1台のDOS/Vマシンに接続し，1つのディスプレイにアプリケーションの実行画面を，もう1つのディスプレイにデバッガ(CodeViewまたはTurbo Debugger)を表示してデバッグを行っている．これもまたDOS/Vマシンのメリットといえるだろう．

さらに，アメリカで発売されている音源ボードであるサウンドブラスタープロとマルチメディアエクステンションを加えて，Windows上でマルチメディア環境を楽しんでいる．まさに，最新の環境が実現されているわけである．

結局，DOS/VマシンのWindows環境下で，DOSおよびWindows用の開発環境を構築すると，その使い勝手は国産のどのパソコンよりも圧倒的に優れているのである．ターゲットマシンはどうあれ，開発マシンにDOS/Vマシンを選択するのは決して間違ったことではないであろう．

DOS/Vマシンのコストパフォーマンス

さて，ここまでDOS/Vマシンを利用した開発環境を見てきたわけだが，このような開発環境を実現するのにいったいどの程度の投資が必要なのだろうか？　いくら優れた開発環境といえども，湯水のごとく金を注ぎ込まなければ実現できないというのでは問題である．

そこで，プログラマに推奨するハードウェアスペックと，実売価格を調べてみることにした．

推奨ハードウェア構成は，次のようなものである．

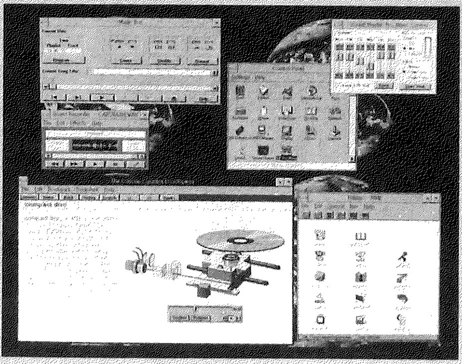
- CPU 80486 33MHz
- HDD最低200MB
- RAM 8MB
- ビデオボード S-VGA
- ディスプレイ 1024×768ドット

アメリカの雑誌広告に掲載されている価格を見ると，この仕様で3000ドルを切る販売価格となっている．つまり，40万円しないのだ．さらに，秋葉原のショップなどで売られているDOS/Vマシンの値段もこの価格に近付いてきている．50万円も出せば，十分開発に使えるフルセットの486マシンが手に入る．開発に使うには少し苦しいがエンドユーザー向けならば十分なスペックのマシンが，ハードディスク，カラーディスプレイつきで20万円を切る価格で発売されているくらいだ．DOS/Vマシン（IBM PC/AT互換

COLUMN　Windows Multimedia Extensions(MME)

数年前からマルチメディアという言葉が，Macintosh関係を中心としてささやかれるようになった．IBM PC，Windowsにおいても，以下のようなマルチメディアの仕様が発表され，この仕様に沿ったマルチメディアパソコンやマルチメディアキットが発売されている．

MPCレベル1
- CPU
  80286（10MHz以上）
- メモリ
  2MB以上
- ディスプレイ
  640×480，16色または256色
- HDD
  30MB以上
- FDD
  3.5インチ，1.44MB
- CD-ROMドライブ
  装備
- WAVE音声
  出力11.025/22.05KHz，8ビットリニア
  入力11.025KHz，8ビットリニア
- MIDI
  入出力ポート，シンセサイザ機能
- ジョイスティック
  装備



Windows MME

メモリ用語集
ハンドル　EMSメモリをアプリケーションから利用する場合，メモリ取得時にハンドルと呼ばれる識別番号を使って読み書きする．

機)のコストパフォーマンスの高さは，国産マシンの比ではない．

さらにビデオボードをより高速なものにすれば，冒頭で紹介したようなX-Windowまがいの日本語Windows環境が実現できる．推奨できるビデオボードは，S3社が開発した86C911というチップセットを搭載した製品だ．このボードは，従来のS-VGAボードの約10倍の処理能力をもっている．これらのビデオボードの値段は，200～600ドル程度だ．

ディスプレイは，やはり17インチくらいのものが欲しくなるが，ちょっと値段がはる．しかし，投資するだけの価値は十分あるだろう．個人輸入をするならば，これだけ揃えても50万円を少し越えるくらいだ．

個人輸入というとビビッてしまう人が多いようだが，注文を英語ですることを除けば，たんなる通信販売である．安いソフトウェアなどからはじめて，慣れたところでハードウェアの個人輸入に挑戦するといいだろう．慣れてくれば，最新の開発ツールなどを手軽に注文できるようになる．実際，へたな国内のパソコンショップに注文するよりも，直接アメリカにオーダーしたほうが早いくらいである．

個人輸入をいやがる人の多くが，英語に苦手意識をもっているようだが，DOS/Vを使いこなすためには，英語のマニュアルを読みこなす程度の力は必要だろう．英語はニガテだという西川氏も，必要な情報はしっかり英語文献から得ている．

DOS/Vマシンには，アメリカなどで発売されているIBM PC/AT用の安価な周辺機器がそのまま利用できる．これは，世界標準パソコンの大きなメリットである．

たとえば，マルチメディアで必要に

COLUMN　マルチメディアキット

IBM PC互換機をマルチメディアマシンにするには，データ転送レートが150KB/s以上のCD-ROMドライブ，Sound Blaster Proなどの音源ボード，Windows Multimedia Extensionsの3つが必要になる．このうちWindows MMEは，実際にはMultimedia Upgrade Kitというキットの形で販売されている．

アメリカなどでは，これら3つをまとめて1パッケージにしたものが，いくつかのメーカーから発売されており，簡単にWindowsでマルチメディアが楽しめるようになっている．DOS/V＋日本語Windowsでも，このアメリカで発売されているキットが利用できるので，マルチメディアを実現することが可能だ．

Windows MMEにはマルチメディアソフトウェアとして，MIDIファイルを演奏する"Jukebox"，CDオーディオを演奏する"Music Box"，さまざまなマルチメディアファイルを再生する"Media Player"，WAVEフォーマットによる録音・再生・編集が可能な"Sound Recorder"などが含まれている．

これらを利用して，パソコンに接続したMIDI楽器をコントロールしたり，コンピュータに演奏をさせながらプログラミングをしたりすることが可能になる．

ここでは，Creative LabsとMedia Visionの2つのキットを紹介する．

【Creative Labsのキット】

- Sound Blaster Pro
- SCD-521内蔵用CD-ROMドライブ（インターフェースはSound Blaster Proのボード上，アクセスタイム390ms）またはSCD-531外付け用CD-ROMドライブ
- Microsoft Windows 3.0＋Multimedia Extensions 1.0＋Creative Music Clips 400（CD-ROM）
- Microsoft Bookshelf for Windows（CD-ROM）
- Creative's SOUNDS Professional audio clips by Prosonus（CD-ROM）
- Sierra On-Line "Jones in the Fast Lane"（CD-ROM）

CD-ROMドライブ内蔵モデル　850ドル
CD-ROMドライブ外付けモデル　950ドル
Creative Labs, Inc.
2050 Duane Avenue
Santa Clara, CA 95054 U.S.A.
FAX(408)986-1777

【Media Visionのキット】

- Media Vision "Pro Audio Spectrum"
- Sony CDU541内蔵用CD-ROMドライブ(SCSI，アクセスタイム380ms)またはSony外付け用CD-ROMドライブ
- Microsoft Windows 3.0＋Multimedia Extensions 1.0（CD-ROM）
- Compton's Multimedia Encyclopedia（CD-ROM）
- Sierra On-Line "Jones in the Fast Lane"（CD-ROM）

CD-ROMドライブ内蔵モデル　995ドル
CD-ROMドライブ外付けモデル　1295ドル
Media Vision
47221 Fremont Blvd
Fremont, CA 94538 U.S.A.
FAX(510)770-8648

メモリ用語集
BMS (Bank Memory Specification)　アイ・オー・データ機器が提唱するIOバンク方式のメモリをアクセスするための仕様

なるCD-ROMドライブなどは，米国で販売されているものをそのまま利用している．日本国内では，SCSI対応のCD-ROMドライブが定価で10万円を切らないが，米国では日本電気製のCD-ROMドライブが599ドルという定価で，さらにディスカウントされて売られている．

最近では，撮影した画像をWindows上で再生するデジタルスチルカメラの廉価版も登場している．

DOS/V プログラミングの参考書

以上で述べてきたように，開発マシンとしてのDOS/Vパソコンには大きな可能性が秘められている．DOS/Vマシンを使いこなせるようになれば，最新のハードウェアを低価格で手に入れ，便利なツールやデータライブラリを用いて快適にプログラムを開発できるのだ．

ここで，IBM PCおよびDOS/Vのプログラミングを学ぶ上で役にたつ参考書を紹介しよう．DOS/Vプログラミングといっても，要はIBM PCのプログラミングである．したがって，まずIBM PC上でのプログラミングについて学ぶ必要がある．

IBM PCを学ぶための良質な参考書としては，「The IBM PC & PS/2」があげられる．これは，日本語版が翔泳社から出ている．この本には，IBM PCでどういうふうにプログラムを作ったらいいのか，というIBM PCプログラミングの基本が書かれている．C言語やアセンブラのサンプルルーチンも豊富に掲載されており，入門者に最適だろう．

IBM PCのビデオシステムに関する日本語の文献としては，やはり翔泳社の「PC & PS/2ビデオシステムプログラマーズガイド」が唯一のものといえる．この本を読めば，IBM PCのビデオ関係の知識は一通り把握できるだろう．

さらに，「PCソースブック」があれば，相当心強いリファレンスとなる．この本は，DOSコマンドからメモリ配置，割り込みデータ，そしてBIOSコールなどIBM PCの開発に必要な情報を1冊にまとめているリファレンス書だ．開発からデバッグまで，この1冊でまかなえる強力なソースブックといえる．

また，洋書には数え切れないほど多くのプログラミング/IBM PC技術解説書が存在している．代表的なものとしては，フェニックス社のBIOS解説書，Addison Wesleyの「UNDOCUMENTED DOS」，「PC INTERRUPTS」，「EXTENDING DOS」，M & T BOOKSの「DOS5: A Developer's Guide」などがあげられる．これらの洋書の中にはフロッピーディスクつきのものも多く，いろいろな場面で役に立つであろう．

最後ににDOS/V特有の問題に関するもっとも詳細なリファレンスは，日本アイ・ビー・エムが発行している「BIOSインターフェース技術解説書」および「DOS J5.0/V技術解説書」

COLUMN　Microsoft Programmer's Library

Microsoft Programmer's Libraryは，米国Microsoft社が発売しているプログラマ向けCD-ROMソフトウェアである．

その内容は，QuickBASIC，MS-C，QuickC，FORTRAN，Assembler，Pascal，QuickPascal，Windows，OS/2などに関するマニュアル，およびMS-DOS Encyclopedia，Programmer's Guide to PC & PS/2 Video Systems，The New Peter Norton Programmer's Guide to the IBM PC and PS/2などの参考文献，さらにはインテル社のCPUに関するリファレンスなどである．

その情報量は膨大なもので，サンプルソースだけで数十メガバイトになるという．これらの情報をエディタなどから簡単に呼び出し，参照したり，そのサンプルソースを自分のソースの中にカット＆ペーストで取り込むことができるのである．プログラマにとって，これほど価値のあるCD-ROMはないであろう．

Microsoft Programmer's Library
Microsoft Corporation
16011 N.E. 36th Way
P.O. Box 97017
Redmond, WA 98073-9717

メモリ用語集
VCPI　DOSエクステンダとEMSエミュレータを共存するための規格．1987年米PharLap社をはじめとする6社によって提唱された．

という2冊のマニュアルである．この2冊で，DOS/V関連のBIOSとファンクションコールは網羅できる．これらのマニュアルは，PSプラザのあるパソコンショップや日本アイ・ビー・エムのサービスセンターで入手可能である．

また，印刷されたマニュアルや技術解説書以外にも，先のマイクロソフトのCD-ROMや，Peter NortonのThe Norton Guidesシリーズのように，メモリに常駐して参照できるデータライブラリの形で発売されているオンラインマニュアルがある．コピーフリーなデータとしても，BIOSのファンクションコール一覧などがあり，パソコン通信を通して配布されている．日本でも，日経MIXをはじめ大手のネットに転載されているので，ダウンロードして利用することができる．

日本と違い，マシンの数と開発者の数が圧倒的に多い米国では，その開発環境を楽にするための工夫も格段に進んでおり，言葉の壁さえ越えられれば情報不足に悩むことはない．むしろ，アメリカのプログラマたちと同時に，いっさいのロスタイムなしに情報を入手でき，さらにその情報を生かしたプログラミングを行えるのは，DOS/Vマシンの大きな利点であるということだ．

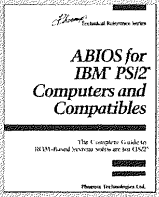
最良の環境で，適切な情報を得てこそ，プログラマはその才能を発揮することができる．最良の環境を低価格で構築できるDOS/Vは，これからのプログラマが注目しなければならない必須の道具だといえるだろう．

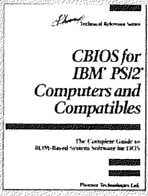
COLUMN　IBM PC BIOS/Interrupt 解説書

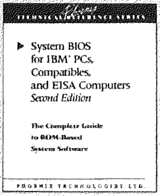
● A BIOS/C BIOS/SYSTEM BIOS

「A BIOS」，「C BIOS」，「SYSTEM BIOS」の3冊は，IBM PC互換機で使われる互換BIOSを開発販売しているフェニックス社による，IBM PC互換機用ROM BIOSの解説書である．実際にBIOSを開発しているフェニックス社の著したものであるから，その信頼性の高さはいうまでもない．

A BIOS　26.95ドル
C BIOS　26.95ドル
SYSTEM BIOS　26.95ドル
Addison Wesley
ISBN 0-201-51805-8
ISBN 0-201-51804-X
ISBN 0-201-57760-7



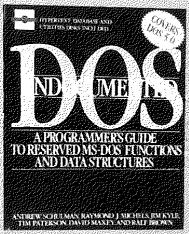




● UNDOCUMENTED DOS

IBM PCの世界で知られるハッカーたちの手になるMS-DOS未公開情報の集大成というべき本．ハイパーテキスト的に検索できるオンラインマニュアルも付属している．

UNDOCUMENTED DOS　44.95ドル
Andrew Schulman, Raymond J.Michels, Jim Kyle, Tim Paterson, David Maxey, and Ralf Brown
Addison Wesley
ISBN 0-201-57064-5

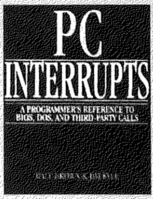


● PC INTERRUPTS

「UNDOCUMENTED DOS」の著者でもあるRalf BrownとJim Kyleによる，DOS，BIOS，サードパーティーの製品（DESQviewやNetWareなど多数）に関するありとあらゆる割り込みの解説書．

これ1冊で，IBM PCのすべての割り込みが理解できる．

PC INTERRUPTS　32.95ドル
Ralf Brown, Jim Kyle
Addison Wesley
ISBN 0-201-57797-6



特集

新しいDOS環境を

探検する

MASM 6.0

MASM6.0の機能と特徴

高級言語に近づくマクロ機能

秋山登

マイクロソフト社のMS-DOS用マクロアセンブラは，MASMの名称で多くのユーザーをもつMS-DOSの標準的なアセンブラである．このマクロアセンブラが久々にバージョンアップされ，PC-9800シリーズ向けのものが昨年「Microsoft Macro Assembler Professional Development System Version 6.0」（以下MASM6.0）として発売された．ここでは，かなり大幅な機能向上のはかられたMASM6.0の機能と特徴について紹介する．

パッケージの内容とインストール

MASM6.0のパッケージは次のものから構成されています.

- 2HDフロッピーディスク　5枚
- インストールガイド（65ページ）1冊
- プログラミングガイド（792ページ）1冊
- リファレンス（240ページ）　1冊
- ユーザー登録カードなど

また, MASM6.0をMS-DOSで実行するのに必要な環境は次のようなものです.

コンピュータ本体：CPU:80286以上

主記憶：640KBのPC-9800およびPC-H98シリーズの各機種（ハイレゾモードを除く）

補助記憶装置：2HDのフロッピーディスクドライブ1台, 空き容量2～8MBのハードディスクドライブ1台

MS-DOS：バージョン3.1以降

もちろん, 増設メモリや拡張メモリあるいはマウスがあれば開発がスムーズになります. なお, 8086やV30をCPUにもつ機種では動作しないので注意が必要です.

インストールはMicrosoft C Professional Development System Version 6.0（以下MS-C6.0）の場合とほぼ同様で, Setupディスクに含まれているSETUP.EXEを実行します. あとは, 画面からインストールのオプションを選択し, 画面に表示される指示に従ってディスクを交換します. MS-DOSのデフォルトの状態でインストールしたときに作成されるサブディレクトリとコピーされるファイルの状況を図1に示します. このとき,

図1　DOSデフォルトのインストール

```
<MASM    >──┬─<BINB    >──┬─H2INC    .EXE  135K
            │             ├─H2INC    .ERR   20K
            │             ├─BIND     .EXE   39K
            │             ├─LINK     .EXE  141K
            │             ├─LIB      .EXE   54K
            │             ├─EXEHDR   .EXE   46K
            │             ├─IMPLIB   .EXE   41K
            │             ├─CVPACK   .EXE   49K
            │             ├─HELPMAKE.EXE    70K
            │             ├─PWBRMAKE.EXE    59K
            │             ├─EXP      .EXE   41K
            │             ├─RM       .EXE   33K
            │             ├─UNDEL    .EXE   42K
            │             └─MASM     .EXE   21K
            ├─<BINR    >──┬─ML       .EXE  244K
            │             ├─ML       .ERR    9K
            │             ├─CV       .EXE  524K
            │             ├─NMAKE    .EXE   79K
            │             ├─QH       .EXE   85K
            │             ├─HIMEM    .SYS   10K
            │             ├─MLX      .EXE   35K
            │             ├─NMK      .COM    3K
            │             ├─PWBED    .EXE  423K
            │             ├─PWB      .COM    3K
            │             ├─PWBROWSE.MXT    35K
            │             ├─PWBMASM .MXT    15K
            │             ├─PWBHELP .MXT    30K
            │             ├─PWBUTILS.MXT    19K
            │             ├─MOUSE    .COM   13K
            │             ├─MOUSE    .SYS   13K
            │             ├─NEW-VARS.BAT     1K
            │             └─NEW-CONF.SYS     1K
            ├─<INCLUDE >──┬─DOS      .INC   11K
            │             ├─CMACROS  .INC   18K
            │             ├─MACROS   .INC    7K
            │             └─PROLOGUE.INC     8K
            ├─<INIT    >────TOOLS    .PRE    5K
            ├─<HELP    >──┬─PWB      .HLP  486K
            │             ├─CV       .HLP  327K
            │             ├─LINK     .HLP  200K
            │             ├─QH       .HLP   39K
            │             ├─UTILERR  .HLP  151K
            │             ├─UTILS    .HLP  345K
            │             ├─ALANG    .HLP  953K
            │             └─ML       .HLP  258K
            ├─<SAMPLES >──┬─SAMPLES  .DOC   10K
            │             │
            │            以下省略
            │            (5サブディレクトリ,
            │             49ファイル, 376K)
            │
            └─README.DOC    15K
```

1クラスタが8KBのハードディスクでは，6MB強の領域を占有します．

図1には含まれませんが，SetupディスクにはREADME.DOC，PACKING.LST，DECOMP.EXEおよびMSKN.COMなどが含まれています．README.DOCは，パッケージの最新情報を格納した文書ファイルで一読しておく必要があります．PACKING.LSTは，フロッピーに格納されて提供されるファイルの一覧表です．DECOMP.EXEは圧縮されたファイルを復元するためのユーティリティです．フロッピーで提供されるファイルは，Setupディスクにある一部のファイルを除いてほとんど圧縮されていますので，そのままコピーしても使用できません．そこでこのユーティリティを使用すれば個別にファイルを復元できますので，必要なシステムファイルをうっかり消してしまったようなときに便利です．MSKN.COMは，ATOK6とATOK7の日本語入力FEPをマイクロソフト社の仕様に合わせるためのTSRプログラムです．

MASM6.0の特徴

MASM6.0の特徴を考えてみると，アセンブラときわめて親しい関係にあるC言語によるプログラミングの普及と開発技法の影響が大であるといわざるをえません．これに加えて，言語処理技術の向上も大きく寄与しているものと思われます．この章では，アセンブリ言語によるプログラミングに新しい方向を示すともいうことのできるMASM6.0の特徴の概要を述べることにします．

システムの構成

MASM6.0のパッケージは，MASM6.0マクロアセンブラ本体ML.EXEとMASM.EXEおよびMLX.EXEのアセンブラドライバ，リンカLINK.EXEとライブラリアンLIB.EXEなどオブジェクトコード関係のユーティリティ，統合開発環境PWBとユーティリティ群，ソースコードレベルのデバッガCV.EXE，オンラインアドバイザとそのデータベース，およびサンプルプログラム群から構成されています．

図1からも明らかなように，統合開発環境PWB，ソースコードレベルデバッガCodeView，オンラインアドバイザなどのユーティリティとデータベースはMS-C6.0と同様の構成になっています．PWBを使用すれば，複数のモジュールから構成されるプログラムのコーディングからデバッグを経て実行にいたるまでを，一元的に管理し遂行することができます．

もし，MS-C6.0をすでにインストールしていれば，重複するものを新しいものと置き換えてディスクの領域を節約することができます．これによって大部分のユーティリティプログラムの領域の重複を回避できます．オブジェクトコードを扱うユーティリティであるLINK.EXEやLIB.EXEあるいはBIND.EXEは機能が増強されている可能性がありますので，ML.EXEのオブジェクトに対してはとくに新しいものを使用しなければなりません．PWBの初期化などに使用するTOOLS.INIファイルのマージも行う必要があります．

MASM6.0のパッケージの中で目新しいソフトウェアは，H2INC.EXEというユーティリティです．これはあとで例をあげますが，C言語のヘッダファイルをアセンブリ言語のインクルードファイルへ変換するためのものです．

オンラインアドバイザ用のデータベースにも新しいものがあります．ALANG.HLPとML.HLPの2つです．ALANG.HLPの内容は，MASM6.0マクロアセンブラの文法の説明，プロセッサ命令の説明，MS-DOSのプログラマーズリファレンス，サンプルプログラムなどです．ML.HLPの内容は，ML.EXE，MASM.EXE，MLX.EXEのコマンドラインの書式とオプションの説明，マクロアセンブラの診断メッセージの解説などです．これらのマニュアルには，文書化された付属のマニュアルにない項目も含まれていますから，PWBあるいはQH.EXEまたはHELPMAKE.EXEでデコードして1度目を通しておいたほうがよいと思われます．

起動法の変更

MASM6.0をコマンドラインから起動する方法は，3種類あります．ML.EXEを直接使用する方法，MASM.EXEを使用する方法，MLX.EXEを使用する方法です．

ML.EXEは，マクロアセンブラ本体とリンカのドライバから構成されています．これは，MS-C5.1や6.0のコンパイラドライバCL.EXEと類似する機能とコマンドラインの書式をもっています．ML.EXEを使用すると

メモリ用語集
プロテクトメモリ　80286以上のCPUで1MB以上の空間にマッピングするメモリのこと．80286CPUでプロテクトモードでしかアクセスできないことからこう呼ばれる．

表1　ML.EXEのオプション(パラメータの書式は省略)

/AT	/Bl	/c	/Cp	/Cu	/Cx	/D	/EP
/F	/Fb	/Fe	/Fl	/Fm	/Fo	/FPi	/FR
/Fr	/Gc	/Gd	/H	/help	/I	/nologo	/Sa
/Sf	/Sg	/Sl	/Sn	/Sp	/Ss	/St	/Sx
/Ta	/w	/W	/WX	/Zd	/Zf	/Zi	/Zm
/Zp	/Zs	/?					

MASM6.0マクロアセンブラの備えているすべてのオプションを使用でき，アセンブルとリンクを1つのコマンドラインから実行させることができます．ML.EXEのオプションの一覧を表1に示します．MS-C6.0のオプションと同じオプションは，CL.EXEに対するオプションと同じような意味をもっています．

MASM.EXEは，MASM5.1用に開発されたソースコードをバッチファイルやメイクファイルを変更することなしにMASM6.0マクロアセンブラでアセンブルするためのアセンブラドライバです．したがって，コマンドラインの書式やオプションはMASM5.1マクロアセンブラのものと同一です．ただし，一部のオプションは無視されるものがあります．表2にMASM.EXEのオプションとそれに対応するML.EXEのオプションの一覧を示します．MASM.EXEはこの表にあげたオプションの変換のほか，MASM5.1と最大限の互換性を保つためのオプション/ZmをML.EXEに渡します．

MLX.EXEは，メモリが不足するようなときにML.EXEをエクステンデッドメモリにロードして実行する場合に使用します．このため，DPMIとXMSの規約を満足するドライバと増設メモリが必要になります．また，MLX.EXEは1度に1つのソースファイルしか処理できないという制限があります．

アセンブル処理時間の短縮

従来からアセンブラで使用されてきたアルゴリズムは，2パス式と呼ばれるものが大半で，以前のバージョンのMASMもこの方式を採用していました．2パス式では，パス1とパス2との2回同じソースプログラムを走査してオブジェクトコードを生成します．このとき，パス1で前方参照の名前(ラベル)を含まないソースコードに対しては直ちにオブジェクトコードを生成できます．しかし，2パス式のアセンブラではこれを行わないで，すべてパス2でオブジェクトコードを生成するようになっています．パス2で再度文字レベルでの走査と解読を行うため，パス1での走査と解読が無駄になってしまいます．

MASM6.0では，このような無駄をできるだけ省くため，以降のパスのためにソースコードを中間コードに落として，パス1のソースコードの走査と解読の結果を利用できるようにしています．そして，この中間コードに対して最適化とオブジェクトコードを生成するためのパスを必要な回数行います．このようなアルゴリズムを採用したため，MASM6.0ではソースに対するパスは1回だけとなり，パスに依存

表2　MASMとMLのオプションの対応

MASMのOPTION	MLのOPTION	ディレクティブ
/A ＊1		.ALPHA
/Bnn ＊2		
/C	/FR	
/D ＊2		
/Dsym[=val]	/Dsym[=val]	
/E	/FPi	
/H	/help	
/HELP		
/Ipath	/Ipath	
/L	/Fl	
/LA	/Fl /Sa	
/ML	/Cp	
/MU	/Cu	
/MX	/Cx	
/N	/Sn	
/P ＊2		OPTION READONLY
/S ＊1		.SEQ
/T	/nologo	
/V ＊2		
/W0	/W0	
/W1	/W1	
/W2	/W2	
/X	/Sx	
/Z ＊2		
/ZD	/Zd	
/ZI	/Zi	
objectfile	/Fofilename	
listingfile	/Flfilename	

＊1：警告が出され無視される
＊2：無条件に無視される

メモリ用語集
プロテクトモード　1MB以上のメモリ空間にアクセスでき，メモリ空間の管理・保護，タスク単位でのプログラム管理などの機能をハードウェアで用意する．

するソースコード(たとえば，IF2や.ERR2など)は無意味になります．また最適化パスを新設したために，前方参照ラベルを原因とするフェーズエラーが発生しないばかりか，コードの質も改善されています．図2-aとbに2パス式とMASM6.0のアルゴリズムの違いを図示します．

MASM6.0ではアルゴリズムの改良により，複数回の最適化パスを実行したり，あとで述べるように機能が以前のバージョンと比べ大幅に増加しているにもかかわらず，アセンブルの処理時間の短縮がはかられています．試しに筆者の手元にあるソースプログラム数本をPC-98XA(80286，8MHz)でアセンブルした結果を表3に示します．このとき使用したコマンドラインは，

```
MASM %1,%1/R/MX; > %1.E
(MASM5.1)
ML /c /Cp %1.ASM > %1.E
(MASM6.0)
```

です．%1はアセンブルするソースプログラムのファイル名で，リンクとリスティングファイルの作成を行わないでオブジェクトプログラムのみを作成しています．処理時間の測定はプロンプトの時刻表示の機能を使用し，秒単位で行いましたのであまり正確とはいえないのですが，この表で見る限り，MASM6.0はMASM5.1に比べて約2割程度処理時間が短縮されていることがわかります．

言語の機能の増強

言語の機能増強はかなり大幅なもの

図2-a　通常の2パス式アセンブル処理

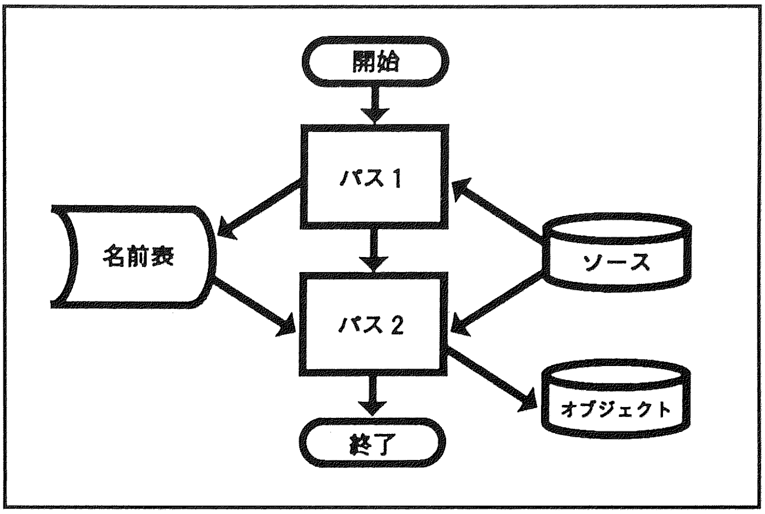


図2-b　MASM6.0のアセンブル処理

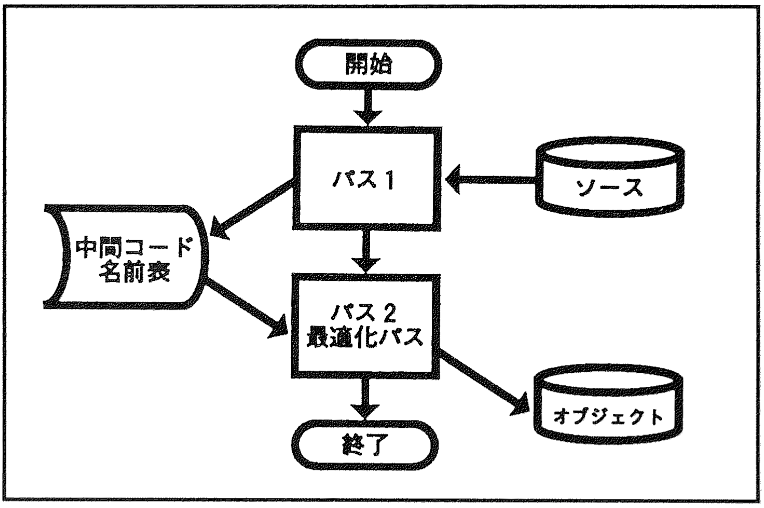


表3　アセンブル処理時間の比較

ソース番号	行数	MASM5.1の処理時間	MASM6.0の処理時間
1	5795	30秒	25秒
2	4020	22秒	17秒
3	1708	10秒	8秒
4	1156	8秒	7秒
5	5020	31秒	25秒
6	2051	17秒	15秒
合計	19750	118秒	97秒

メモリ用語集
ページフレーム（Page Frame）　論理ページをマッピングするための物理ページの空間を指す．EMSメモリでは64KBのページフレームが用意される．

があります．以下に増強された機能の要点をあげます．具体的な例は第3章で説明します．

・最適化機能の強化

複数の最適化パスを設けたことによって不要なNOP命令を生成しなくなりました．そのほか，一部の命令に対してより効率のよいコードを生成するようになっています．

・486CPUのサポート

486のすべての命令がサポートされています．そのほか386以上の命令で，アドレスサイズプリフィックスやオペランドサイズプリフィックスをDBディレクティブで書かなくてもよいようにニモニックが増やされています．

・FLATメモリモデルのサポート

386以上のプロセッサでは，セグメントが1つで，セグメントサイズに制限のないFLATメモリモデルのアセンブルがサポートされています．このメモリモデルは，MS-DOSやOS/2などによってサポートされていませんが，新しいOSを開発したり，スタンドアローンのシステムの開発には有効と思われます．

・構造化プログラミング手法の導入

実行時の選択や繰り返しの処理をディレクティブ(定義済みマクロ)を使用して簡潔に書くことができます．このディレクティブ群によってソースコードからほとんどのジャンプ系命令とジャンプ先のラベルを取り除くことができます．

・マクロ機能の強化

マクロの文法は，かなり改訂されより使いやすくなっています．また，従来のマクロ(マクロ手続き)のほかに，マクロ関数が新設されています．

・プロジェクト管理機能の強化

C言語で使用されるヘッダファイル(.Hファイル)によるグローバルな資源の管理とほぼ同様な管理が行えるよう，いくつかの便利なディレクティブが追加されています．また，このような管理を容易にするため，前に述べたH2INC.EXEがユーティリティとして付属しています．

・高級言語との親和性の向上

高級言語に対応するデータの型と構造が取り入れられています．高級言語の整数に対応する型には符号付きと符号なしの区別をつけることができます．構造体に関する文法は改訂され，新たに共用体が使用できるようになっています．高級言語の手続き(関数)の呼び出しと後始末の命令群を生成するディレクティブも追加されています．

・その他

アセンブルからリンクにいたる名前は，最大247文字まで認識されます．このほか，レジスタの用途のチェック，ディレクティブのキーワードの一貫性を保つための変更，多数のMASM 5.1互換オプションなどがあげられます．

MASM6.0の新しい機能

この章では，MASM6.0の新しい機能をプログラミングの例を交えて紹介します．ほとんどの機能はまったく新しいものか，従来の機能との互換性を維持しながら機能を拡張したものかのいずれかですが，中には従来の機能と互換性が失われたものもあります．

ディレクティブ

アセンブラの機能はディレクティブを見るとよくわかります．表4にMASM6.0で新たに追加されたディレクティブを示します．表4の中で()

表4　新しいディレクティブ(()内のディレクティブは同じ効果を持つ)

.486	.486P	.BREAK
.CONTINUE	.DOSSEG (DOSSEG)	.ELSE
.ELSEIF	.ENDIF	.ENDW
.EXIT	.IF	.LISTALL
.LISTIF (.LFCOND)	.LISTMACRO (.XALL)	.LISTMACROALL (.LALL)
.NO87	.NOCREF (.XCREF)	.NOLIST (.XLIST)
.NOLISTIF (.SFCOND)	.NOLISTMACRO(.SALL)	.REPEAT
.STARTUP	.UNTIL	.UNTILCXZ
.WHILE	ECHO (%OUT)	EXTERN (EXTRN)
EXTERNDEF	FOR (IRP)	FORC (IRPC)
GOTO	INVOKE	OPTION
POPCONTEXT	PROTO	PUSHCONTEXT
REPEAT (REPT)	STRUCT (STRUC)	SUBTITLE (SUBTTL)
TEXTEQU	TYPEDEF	UNION
WHILE		

メモリ用語集
マッピング(Mapping)　拡張メモリの論理ページを物理ページでアドレッシングするための処理．

に入っているディレクティブは，同様の機能を果たす従来のディレクティブで，MASM6.0でも有効です．これらのディレクティブは，高級言語のユーザーにとってわかりやすいものになっています．

従来のディレクティブでサポートされなくなったディレクティブには．MSFLOATがあります．また，パスに依存する.ERR1, IF1, ELSEIF1の各ディレクティブは，それぞれ.ERR, IF, ELSEIFと同じ意味をもち，.ERR2, IF2, ELSEIF2はエラーとなります．

ここで表4にはありませんが，従来からのディレクティブで機能が拡張されたディレクティブで重要と思われるものを2つ紹介しておきます．

・.MODEL

.MODELディレクティブは，マイクロソフト社の言語規約に基づく簡略化されたセグメントの使用を宣言するディレクティブです．メモリモデルのほか，モジュール全体を通してデフォルトに使用されるそのほかの属性が指定できます．

・.MODELメモリモデル[,言語][,OS][,スタックの距離]

メモリモデルは，TINYからHUGEに加えてFLATが指定できます．言語は，モジュール内で定義されるパブリックシンボルと手続き名の命名規約と手続きの呼び出し規約に使用され，C，BASIC，FORTRAN，Pascalのほか，SYSCALLとSTDCALLが指定可能です．OSはこのモジュールに制御を渡すオペレーティングシステムの名前で，OS_DOSまたはOS_OS2を指定します．この指定は，OSから制御が渡されたときの開始コードを作成する.STARTUPディレクティブとOSに制御を戻すときの終了コードを作成する.EXITディレクティブに影響を与えます．スタックの距離は，DS == SSと仮定できるときNEARSTACKを，DS!= SSと仮定するときFARSTACKを指定します．

・ASSUME

ASSUMEディレクティブは，すべてのレジスタの使用法をチェックできるように，次のように書式が拡張されています．

ASSUMEセグメントレジスタ:セグメント位置,...
ASSUMEデータレジスタ:修飾型,...
ASSUMEレジスタ:ERROR,...
ASSUME [レジスタ:]NOTHING [,レジスタ:NOTHING],...

上の書式の内，最初の書式は従来から使用されているものです．ただし，セグメント位置にはFLATが指定可能になりました．次の3つの書式はセグメントレジスタに加えて，データレジスタにも適用可能です．たとえば，

```
ASSUME BX:ERROR
..........
MOV BX,20
```

のようなソースコードがあると，診断メッセージが表示されます．修飾型については，「データ型」の節で述べます．

プロセッサ命令

.486あるいは.486Pディレクティブによって486CPUの命令がすべてサポートされます．このほか，オペランドのサイズを明確にするために新たなニモニックが追加されています．表5に新しく追加されたニモニックと486CPUのためのレジスタ記号を示します．逆にサポートされなくなった命令にはESCがあります．

表5　新しいプロセッサ命令とレジスタ記号

プロセッサ命令：()内は使用できる最下位のプロセッサ		
BSWAP (486)	CMPXCHG (486)	FLDENVD (387)
FLDENVW (8087)	FNSAVED (387)	FNSAVEW (8087)
FNSTENVD (387)	FNSTENVW (8087)	FRSTORD (387)
FRSTORW (8087)	FSAVED (387)	FSAVEW (8087)
FSTENVD (387)	FSTENVW (8087)	INVD (486)
INVLPG (486)	IRETDF (386)	IRETF (386)
LOOPD (386)	LOOPED (386)	LOOPEW (8086)
LOOPNED (386)	LOOPNEW (8086)	LOOPNZD (386)
LOOPNZW (8086)	LOOPW (8086)	LOOPZD (386)
LOOPZW (8086)	POPAW (186)	PUSHAW (186)
PUSHD (386)	PUSHW (186)	WBINVD (486)
XADD (486)		
レジスタ記号：()内は使用できる最下位のプロセッサ		
TR3 (486)	TR4 (486)	TR5 (486)

メモリ用語集
メモリウィンドウ（Memory Window）　ページフレームの別の呼び方．

定義済みの名前

表6は，MASM6.0で追加された定義済みの名前です．このうち，@CatStr，@InStr，@SizeStrおよび@SubStrはマクロ関数の名前で，CATSTR，INSTR，SIZESTR，およびSUBSTRディレクティブに対応しています．@Environもマクロ関数で，環境変数の名前を引数にとり，その環境変数に設定されている値を返します．@Dateと@Timeは現在の日付と時刻を返すテキストマクロです．@FileCurはアセンブルしている拡張子を含むファイル名を，@Lineは現在の行番号をそれぞれ返します．@stackは，スタックセグメントを含むグループ名を返します．@Interfaceと@Modelは，それぞれ現在の言語規約とメモリモデルをコード化した数値を返します．

リスト1は，アセンブルの時点での定義済みの名前をコンソールに表示する簡単なプログラムです．このプログラムの中でCOSTRは，DS:SIから始まり00Hで終わる文字列をコンソールに表示する手続きです．2つのマクロSTRLBLとDEFSTRは，ラベルに定義されている値を文字列としてメモリに定義するため使用されています．このマクロを使用すれば，このほかに数値や文字列として定義されているラベルの値を文字列として定義することができます．

表6　新しい定義済みの名前

@CatStr	@Date	@Environ
@FileCur	@InStr	@Interface
@Line	@Model	@SizeStr
@SubStr	@Time	@stack

リスト1　EXSYM.ASM

```
.MODEL SMALL, SYSCALL, OS_DOS, FARSTACK
EXTERN COSTR:NEAR
;
; ラベルの値を文字列に変換してバイト列として定義する
;    変換するラベル:LBL(可変個)
;
STRLBL  MACRO LBL:VARARG
        FOR LBLV,<LBL>
            DEFSTR %LBLV
        ENDM
    ENDM
;
; 文字列をバイト列として定義する
;    定義する文字列:STRV
;
DEFSTR  MACRO STRV
    BYTE '&STRV&'
    ENDM
;
LF  EQU 0AH
CR  EQU 0DH
;
.CODE
;
; 定義済みシンボルのアセンブル時の値を表示する
;
EXSYM PROC
    .STARTUP
    LEA SI,OFFSET MSG
    CALL COSTR
    .EXIT 0
EXSYM ENDP
;
.DATA
MSG BYTE CR,LF,'@Model =                       '
    STRLBL @Model
    BYTE CR,LF,'@stack =                       '
    STRLBL @stack
    BYTE CR,LF,'@Environ(COMSPEC) = '
?ENV    TEXTEQU @Environ(COMSPEC)
    STRLBL ?ENV
    BYTE CR,LF,'@Interface =                   '
    STRLBL @Interface
    BYTE CR,LF,'@Date =                        '
    STRLBL @Date
    BYTE CR,LF,'@Time =                        '
    STRLBL @Time
    BYTE CR,LF,'@FileCur =                     '
    STRLBL @FileCur
    BYTE CR,LF,'@Line =                        '
    STRLBL @Line
    BYTE CR,LF,0
;
.STACK
;
END
```

図3

```
FPTR TYPEDEF FAR PTR    ; void _far * FPTR
IPTR TYPEDEF PTR SWORD  ; int * IPTR
```

メモリ用語集
メモリ保護　マルチタスクにおいて，各タスクのアプリケーションがほかのアプリケーションのコードやデータの破壊を防ぐこと．

データ型

MASM6.0のデータ型には組み込み型と修飾型の２つがあります．組み込み型には高級言語との間の型チェックを厳重に行うため，SBYTE, SWORD, SDWORD, REAL4, REAL8, およびREAL10が追加されています．SBYTE, SWORD, およびSDWORDは，それぞれBYTE, WORD, およびDWORDの符号付き整数の型に対応します（BYTE, WORD, およびDWORDは，符号なし整数を意味します）．REAL4, REAL8, およびREAL10は，それぞれDWORD, QWORD, およびTBYTEの浮動小数点型に対応します．

修飾型はMASM6.0の新しい概念で，上記の組み込み型とTYPEDEFディレクティブで宣言される型などです．TYPEDEFディレクティブは，C言語のtypedef文と同様の働きをします．図３にTYPEDEFディレクティブの例をあげます．";"の右側は，対応するC言語の型を示します．

修飾型は，型名の使用できるあらゆる文脈で使用することができます．また，データを定義するDB, DW, DD, DF, DQ, およびDTの各ディレクティブのかわりに使用することもできます．

データ構造

MASMの以前のバージョンでは，データ構造として構造体とレコードが使用できました．MASM6.0ではこれに加えて共用体（UNION）が使用できます．レコードの扱いはMASM5.1と同じですが，構造体の扱いは次のように変更されています．

リスト２ EXDIR.ASM

```
INCLUDE EXSUB.INC
;
MSDOS   EQU 21H       ; MS-DOS機能呼出ベクトル番号
DOSSDTA EQU 1AH       ; DTAの設定(DOS機能番号)
DOSFFST EQU 4EH       ; 最初のファイル検索(DOS機能番号)
DOSFNXT EQU 4FH       ; 次のファイル検索(DOS機能番号)
;
DIRDAT STRUCT               ; ディレクトリデータの書式
    RESVD BYTE  21 DUP(?)        ; 予約
    ATTR  BYTE  ?               ; 属性
    MTIME WORD  ?               ; 最終更新時刻
    MDATE WORD  ?               ; 最終更新日付
    FSIZE DWORD ?               ; ファイルの容量
    FNAME BYTE  13 DUP(?)        ; ファイル名
    FILLR BYTE  85 DUP(?)        ; 予備
DIRDAT ENDS
TIMEREC RECORD HR:5, MI:6, SC:5 ; 最終更新時刻の書式
DATEREC RECORD YR:7, MO:4, DY:5 ; 最終更新日付の書式
;
.CODE
;
; カレントディレクトリの表示
;
TST02 PROC
    .STARTUP
    MOV AX,DS
    MOV ES,AX
    MOV DX,OFFSET DIRBUF     ; DTAの設定
    MOV AH,DOSSDTA
    INT MSDOS
    MOV DX,OFFSET DIRSRC     ; 最初のファイルの検索
    MOV CX,003FH           ; すべての属性
    MOV AH,DOSFFST
    INT MSDOS
    .IF !CARRY?
        CALL DIRDSP
        .WHILE 1
            MOV AH,DOSFNXT        ; 次のファイルの検索
            INT MSDOS
            .BREAK .IF CARRY?
            CALL DIRDSP
        .ENDW
    .ENDIF
    .EXIT 0
TST02 ENDP
;
; DIRBUFの内容をOUTBUFで整理して表示する
;
DIRDSP PROC NEAR PRIVATE
    MOV SI,OFFSET DIRBUF.FNAME  ; ファイル名
    MOV DI,OFFSET OUTBUF
    CLD
    LODSB
    STOSB
    .IF AL=='.'
        MOV CX,11
        .WHILE 1
            LODSB
            .BREAK .IF AL==00H
            STOSB
            DEC CX
        .ENDW
        MOV AL,' '
        REP STOSB
    .ELSE
        MOV CX,7
        .WHILE 1
            LODSB
            .BREAK .IF AL==00H || AL=='.'
            .IF AL==05H
                MOV AL,0E5H
            .ENDIF
            STOSB
            DEC CX
        .ENDW
        .IF AL!=00H
            MOV AL,' '
            REP STOSB
            MOV AL,DIRBUF.ATTR
            AND AL,08H
            .IF ZERO?
                MOV AL,'.'
                STOSB
                MOV CX,3
            .ELSE
                MOV CX,4
            .ENDIF
            .WHILE 1
                LODSB
                .BREAK .IF AL==00H
                .IF AL==05H
                    MOV AL,0E5H
                .ENDIF
                STOSB
                DEC CX
            .ENDW
            MOV AL,' '
            REP STOSB
        .ELSE
            ADD CX,4
```

メモリ用語集

メモリマネージャ（Memory Manager） 拡張メモリやプロテクトメモリなどを複数のアプリケーションから利用する際，競合が起きないようメモリを管理するプログラム．

①構造体は，ほかの構造体や共用体とネストすることが許されます．

②構造体のフィールド名はラベルや，ほかの構造体のフィールド名と同じでもかまいません．

③構造体のフィールドへの参照は，構造体名で直接的に，あるいは変数名などを通して間接的に完全に限定しなければなりません．

④構造体のフィールドに関する演算子は，"." 以外のものは使用できません．逆に，"." は，"+" 演算子のかわりに使用することもできなくなりました．

⑤構造体の実体を初期化する際の<>は，{}を使用しても同じ効果をもちます．

共用体は，構造体と同じ構文で宣言することができます．次の例は，構造体STRUCEXと共用体UNEXをネストして宣言する例です．ネストする場合，内側にあるENDSには構造体や共用体の名前を付けません．

```
STRUCEX STRUCT
    F1 BYTE 'ABCD'
    F2 WORD ?
    UNEX UNION
        U1 WORD 0
        U2 BYTE ?
    ENDS
STRUCEX ENDS
```

リスト2は，構造体とレコードの使用例です．このプログラムは，カレントディレクトリにあるすべてのファイルの情報を表示します．このプログラムは多少複雑ですが，8行目から18行目にある構造体とレコードの宣言と，199行目にある構造体の定義，50行目と120行目にある構造体の参照，130行目にあるレコードのビットフィールドの参照などを注目してください．EXSUB.INCというファイルをインクルードしていますが，この中に簡略化セグメントの使用の宣言と，LTOA，ULTOAおよびCOSTRの

```
101:             MOV AL,' '
102:             REP STOSB
103:         .ENDIF
104:     .ENDIF
105:     MOV AL,' '
106:     STOSB
107:     MOV AL,DIRBUF.ATTR
108:     AND AL,18H
109:     .IF !ZERO?
110:         AND AL,10H
111:         .IF !ZERO?             ; サブディレクトリ
112:             MOV SI,OFFSET DIRATR
113:             MOV CX,SIZEOF DIRATR-1
114:         .ELSE                  ; ボリュームラベル
115:             MOV SI,OFFSET VOLATR
116:             MOV CX,SIZEOF VOLATR-1
117:         .ENDIF
118:         REP MOVSB
119:     .ELSE                      ; ファイルの容量
120:         MOV AX,WORD PTR DIRBUF.FSIZE
121:         MOV DX,WORD PTR DIRBUF.FSIZE+2
122:         MOV BX,10
123:         MOV CX,8
124:         INVOKE LTOA
125:         DEC DI
126:     .ENDIF
127:     MOV AL,' '
128:     STOSB
129:     MOV AX,DIRBUF.MDATE        ; 最終更新日付（年）
130:     MOV CL,WIDTH MO+WIDTH DY
131:     SHR AX,CL
132:     ADD AX,80                  ; 西暦に変換
133:     .IF AX>99
134:         SUB AX,100
135:     .ENDIF
136:     XOR DX,DX
137:     MOV BX,10
138:     MOV CX,2
139:     INVOKE ULTOA
140:     DEC DI
141:     MOV AL,'-'
142:     STOSB
143:     MOV AX,DIRBUF.MDATE        ; 最終更新日付（月）
144:     AND AX,MASK MO
145:     MOV CL,WIDTH DY
146:     SHR AX,CL
147:     XOR DX,DX
148:     MOV BX,10
149:     MOV CX,2
150:     INVOKE ULTOA
151:     DEC DI
152:     MOV AL,'-'
153:     STOSB
154:     MOV AX,DIRBUF.MDATE        ; 最終更新日付（日）
155:     AND AX,MASK DY
156:     XOR DX,DX
157:     MOV BX,10
158:     MOV CX,2
159:     INVOKE ULTOA
160:     DEC DI
161:     MOV AL,' '
162:     STOSB
163:     MOV AX,DIRBUF.MTIME        ; 最終更新時刻（時）
164:     MOV CL,WIDTH MI+WIDTH SC
165:     SHR AX,CL
166:     XOR DX,DX
167:     MOV BX,10
168:     MOV CX,2
169:     INVOKE ULTOA
170:     DEC DI
171:     MOV AL,':'
172:     STOSB
173:     MOV AX,DIRBUF.MTIME        ; 最終更新時刻（分）
174:     AND AX,MASK MI
175:     MOV CL,WIDTH SC
176:     SHR AX,CL
177:     XOR DX,DX
178:     MOV BX,10
179:     MOV CX,2
180:     INVOKE ULTOA
181:     DEC DI
182:     MOV AL,':'
183:     STOSB
184:     MOV AX,DIRBUF.MTIME        ; 最終更新時刻（秒）
185:     AND AX,MASK SC
186:     ADD AX,AX                  ; 1秒単位へ
187:     XOR DX,DX
188:     MOV BX,10
189:     MOV CX,2
190:     INVOKE ULTOA
191:     MOV SI,OFFSET OUTBUF
192:     INVOKE COSTR
193:     MOV SI,OFFSET CRLF
194:     INVOKE COSTR
195:     RET
196: DIRDSP ENDP
197: ;
198: .DATA?
199: DIRBUF  DIRDAT  <>         ; ファイル検索用バッファ
200: OUTBUF  BYTE    128 DUP(?) ; ファイルの情報表示用バッファ
201: ;
202: .CONST
203: CRLF    BYTE    0DH,0AH,00H ; 復帰改行
204: DIRSRC  BYTE    '*.*',00H   ; 検索するファイル
205: VOLATR  BYTE    ' < VOL >',00H ; ボリュームラベルの表示用
206: DIRATR  BYTE    ' < DIR >',00H ; サブディレクトリの表示用
207: ;
208: .STACK
209: ;
210: END
```

メモリ用語集

メモリモデル　8086CPUのメモリ空間は64KB単位のセグメントで表現される．そのためデータ領域やコード領域が64KBを堺にして複数のメモリモデルが存在する．

外部記号が宣言されています．LTOAは，符号付きの4バイト整数を文字列に，ULTOAは符号なしの4バイト整数を文字列に変換する外部手続きです．COSTRはリスト1と同じものです．このプログラムの中の手続きはTST02とDIRDSPの2つです．実行はTST02の最初の行からはじまり，この中でMS-DOSの機能を呼び出して，カレントディレクトリを検索します．ファイルが見つかれば，DIRBUFに設定された情報をもとにDIRDSPがその情報を整理してコンソールに表示します．

式の評価と演算子

マクロや流れ制御に関するディレクティブ以外のプロセッサ命令やディレクティブに使用される式の評価法での重要な変更点は，次のようなものです．

①式はプロセッサの指定にかかわらず，つねに32ビット精度で評価されます．
②基数が10でない定数は符号なしとみなされます．たとえば，"0FFFFH GT -1" の" 0FFFFH" は65535と解釈され，式は真になります．
③" DISP-10[BP]" のような式は，" (DISP-10)[BP]" ではなく" DISP-(10[BP])" と解釈され，エラーとなります．同様に" -10[BP]" の式もエラーとなります．

表7　新しい演算子

HIGHWORD	LENGTHOF	LOWWORD
LROFFSET	OPATTR	SIZEOF

図4

```
EXPRM MACRO PRM1:REQ, PRM2:=<AX>, PRM3:VARARG
...........
ENDM
```

④定数の基数を表す接尾辞が10進数の場合，"D" から "T" へ，2進数の場合，"B" から "Y" へ変更されています．

MASM6.0では，表7に示す演算子が追加されています．HIGHWORDとLOWWORDは直後にイミーディエット式をとり，それぞれ32ビット値の上位16ビットと下位16ビットを返します．同様に，従来のHIGHとLOW演算子もイミーディエット式のほかは対象として許されなくなりました．メモリ式を対象とする場合は，次の例のようにOFFSET演算子が必要になります．

```
MOV AL,LOW OFFSET @F
MOV AH,HIGH OFFSET @F
..........
@@:
```

LENGTHOF演算子は，対象となるラベルに定義されたデータの要素数を返します．また，SIZEOF演算子は，C言語のsizeof演算子と同様に対象となるラベルに定義されたデータのバイト数を返します．OPATR演算子は，MASM5.1の.TYPE演算子と同様に使用できますが，ビット8～10に言語の情報を追加して返します．LROFFSET演算子はWindowsのプログラミングに使用される特殊な演算子で，この演算子で得られたOFFSET値は，リンカによってリロケータブルな項目として実行ファイルに書込まれます．

マクロ機能

マクロの機能は，かなり大幅な変更が加えられています．その要点をあげると以下のようなものです．

①マクロ引数の区切りが，スペース，タブ，コンマであったのがコンマのみになりました．これによってたとえば，" WORD PTR [BX]" のような引数をマクロに渡すのが容易になります．
②マクロ引数に必須，可変長，デフォルト値のタグを付けることが可能になりました．図4の例では，PRM1は必須の引数であり，PRM2が指定されないときはデフォルト値としてAXを採用し，PRM3以降の引数は可変長であるマクロの定義を示します．
③WHILE，GOTOの2つの新しいディレクティブが追加されています．また，テキストマクロのためにTEXTEQUというEQUディレク

メモリ用語集
UMB（Upper Memory Block）640KBから1MBのROM空間を指す．ここにRAMをマッピングし，TSRやデバイスドライバを移動させ，メインメモリを広くすることが可能

ティブに代わるディレクティブが追加されました.

④置換演算子"&"の使用法と取り扱いが変更されました. 基本的な使用法は, "&PARAMETER&"です. ただし, 状況によってはどちらか一方の"&"は省略できます. なお, マクロ引数の中にある"&"は, マクロのネストレベルに関係なく置換によってのみ取り除かれます.

⑤マクロ関数が新設されました. マクロ関数の定義は, 通常のマクロと1点を除いて変わりません. その1点とはマクロを抜け出すとき式を伴ったEXITMディレクティブで抜け出す必要があることです. 式の値は, つねにテキスト値として呼び出し元に返されます.

リスト3は, マクロとマクロ関数の例です. WCALC2は通常のマクロで5行目にある形式の式を文字列として受け取り, それをプロセッサ命令に展開します. ?Token, ?Tnext, および?Skpblはマクロ関数です. ?Tokenは, WCALC2から呼び出され, 引数の文字列の先頭にある要素あるいは演算子を返します. ?TnextもWCALC2から呼び出され, 引数の文字列の先頭にある要素あるいは演算子を取り除きます. ?Skpblは, ?Tokenと?Tnextから呼び出され, 引数の文字列の先頭にあるスペースとタブ文字を取り除きます.

このプログラムには, EXMACとDSPRSLTの2つの手続きがあります. EXMACの先頭から実行がはじまり, 最初にWCALC2マクロを呼び出して240と35の間で何種類かの計算を行い結果を作業領域に格納します. それからDSPRSLTを呼び出して結果をコンソールに表示します. DSPRSLTは, DS:SIではじまる文字列を表示し, 計算の結果であるAXの数値を10進数, 16進数および2進数の順序で

リスト3 EXMAC.ASM

```
  1: INCLUDE EXSUB.INC
  2: ;
  3: ; WORD型の要素の からなる2項式を機械命令に変換するマクロ
  4: ;    2項式の文字  ?Estr
  5: ;    2項式:<要素1>=<要素2><演算子><要素3>
  6: ;
  7: WCALC2  MACRO ?Estr:REQ
  8:     ??Wtk0 TEXTEQU <?Estr>
  9:     ??Wtk1 TEXTEQU ?Token(<%??Wtk0>)
 10:     ??Wtk0 TEXTEQU ?Tnext(<%??Wtk0>)
 11:     ??Wtk2 TEXTEQU ?Token(<%??Wtk0>)
 12:     ??Wtk0 TEXTEQU ?Tnext(<%??Wtk0>)
 13:     ??Wtk3 TEXTEQU ?Token(<%??Wtk0>)
 14:     ??Wtk0 TEXTEQU ?Tnext(<%??Wtk0>)
 15:     ??Wtk4 TEXTEQU ?Token(<%??Wtk0>)
 16:     ??Wtk0 TEXTEQU ?Tnext(<%??Wtk0>)
 17:     ??Wtk5 TEXTEQU ?Token(<%??Wtk0>)
 18:     ??Wtk0 TEXTEQU ?Tnext(<%??Wtk0>)
 19:     IFNB ??Wtk0
 20:         ECHO 余計な要素があります(WCALC).
 21:         EXITM
 22:     ENDIF
 23:     IFDIFI ??Wtk3,<AX>
 24:         MOV AX,??Wtk3
 25:     ENDIF
 26:     IFIDN   ??Wtk4,<&>
 27:         AND AX,??Wtk5
 28:     ENDIF
 29:     IFIDN   ??Wtk4,<|>
 30:         OR AX,??Wtk5
 31:     ENDIF
 32:     IFIDN   ??Wtk4,<^>
 33:         XOR AX,??Wtk5
 34:     ENDIF
 35:     IFIDN   ??Wtk4,<+>
 36:         ADD AX,??Wtk5
 37:     ENDIF
 38:     IFIDN   ??Wtk4,<->
 39:         SUB AX,??Wtk5
 40:     ENDIF
 41:     IFIDN   ??Wtk4,<*>
 42:         MOV CX,??Wtk5
 43:         MUL CX
 44:     ENDIF
 45:     IFIDN   ??Wtk4,</>
 46:         MOV CX,??Wtk5
 47:         MOV DX,0
 48:         DIV CX
 49:     ENDIF
 50:     IFIDN ??Wtk4,<¥>
 51:         MOV CX,??Wtk5
 52:         MOV DX,0
 53:         DIV CX
 54:     ENDIF
 55:     IFIDN ??Wtk2,<=>
 56:         IFIDN ??Wtk4,<¥>
 57:             IFDIFI ??Wtk1,<DX>
 58:                 MOV ??Wtk1,DX
 59:             ENDIF
 60:         ELSE
 61:             IFDIFI ??Wtk1,<AX>
 62:                 MOV ??Wtk1,AX
 63:             ENDIF
 64:         ENDIF
 65:     ENDIF
 66:     ENDM
 67: ;
 68: ; 文字列の先頭のトークンを返すマクロ関数
 69: ;    対象とする文字列:?Estr
 70: ;    先頭のトークン:戻値
 71: ;
 72: ?Token  MACRO   ?Estr:REQ
 73:     ??Tstr1 TEXTEQU ?Skpbl(?Estr)
 74:     ??Tstr2 TEXTEQU <>
 75:     IFNB ??Tstr1
 76:         ??Tstr2 TEXTEQU @SubStr(<%??Tstr1>,1,1)
 77:         ??Tmat = 0
 78:         FORC ??Twrk1,<=+-*/¥&|^~>
 79:             IFIDN ??Tstr2,<??Twrk1>
 80:                 ??Tmat = 1
 81:                 EXITM
 82:             ENDIF
 83:         ENDM
 84:         IF ??Tmat
 85:             EXITM ??Tstr2
 86:         ENDIF
 87:         ??Tstr2 TEXTEQU <>
 88:         %FORC ??Twrk1,<??Tstr1>
 89:             ??Tmat = 0
 90:             FORC ??Twrk2,<  =+-*/¥&|^~>
 91:                 IFIDN <??Twrk1>,<??Twrk2>
 92:                     ??Tmat = 1
 93:                     EXITM
 94:                 ENDIF
 95:             ENDM
 96:             IF ??Tmat
 97:                 EXITM
 98:             ENDIF
 99:             ??Tstr2 TEXTEQU @CatStr(<%??Tstr2>,<??Twrk1>)
100:         ENDM
101:     ENDIF
102:     EXITM ??Tstr2
```

メモリ用語集

UMBローダ (UMB Loader)　UMB領域にTSRやデバイスドライバを組み込むためのプログラム.

```
    ENDM
;
; 文字列の先頭のトークンを除去するマクロ関数
;   対象とする文字列:?Estr
;   先頭のトークンを除去した文字列:戻値
;
?Tnext  MACRO   ?Estr:REQ
    ??Tstr1 TEXTEQU ?Skpbl(?Estr)
    IFNB ??Tstr1
        ??Tstr2 TEXTEQU @SubStr(<%??Tstr1>,1,1)
        ??Tmat = 0
        FORC ??Twrk1,<=+-*/¥&|^~>
            IFIDN ??Tstr2,<??Twrk1>
                ??Tmat = 1
                EXITM
            ENDIF
        ENDM
        IF ??Tmat
            EXITM @SubStr(<%??Tstr1>, 2)
        ENDIF
        ??Tpos = 1
        ??Tcnt = @SizeStr(<%??Tstr1>)
        %FORC ??Twrk1,<??Tstr1>
            ??Tmat = 0
            FORC ??Twrk2,<  =+-*/¥&|^~>
                IFIDN <??Twrk1>,<??Twrk2>
                    ??Tmat = 1
                    EXITM
                ENDIF
            ENDM
            IF ??Tmat
                EXITM
            ENDIF
            ??Tpos = ??Tpos+1
        ENDM
        IF ??Tpos GT ??Tcnt
            EXITM <>
        ELSE
            EXITM @SubStr(<%??Tstr1>, ??Tpos)
        ENDIF
    ENDIF
    EXITM <>
    ENDM
;
; SPACEとTABを文字列から取り除くマクロ関数
;   対象とする文字列:?Estr
;   処理された文字列:戻値
;
?Skpbl  MACRO   ?Estr:REQ
    ??Spos = 1
    FORC ?Swrk,<?Estr>
        IFDIF <?Swrk>,< >
            IFDIF <?Swrk>,<	>
                EXITM
            ENDIF
        ENDIF
        ??Spos = ??Spos+1
    ENDM
    EXITM   @SubStr(<?Estr>,??Spos)
    ENDM
;
.CODE
;
; 各種演算の結果を表示する
;
EXMAC PROC NEAR
    .STARTUP
    MOV AX,DS
    MOV ES,AX
    MOV SI,XX           ; SI=XX
    MOV DI,YY           ; DI=YY
    WCALC2 XYAND=SI&DI
    WCALC2 XYIOR=SI|DI
    WCALC2 XYEOR=SI^DI
    WCALC2 XYADD=SI+DI
    WCALC2 XYSUB=SI-DI
    WCALC2 XYMUL=SI*DI
    WCALC2 XYDIV=SI/DI
    WCALC2 XYREM=SI¥DI
    MOV AX,XX           ; XXの値を表示
    MOV SI,OFFSET MSGX
    CALL DSPRSLT
    MOV AX,YY           ; YYの値を表示
    MOV SI,OFFSET MSGY
    CALL DSPRSLT
    MOV SI,OFFSET CRLF
    INVOKE COSTR
    MOV AX,XYAND            ; XX AND YYの値を表示
    MOV SI,OFFSET MSGAND
    CALL DSPRSLT
    MOV AX,XYIOR            ; XX OR YYの値を表示
    MOV SI,OFFSET MSGIOR
    CALL DSPRSLT
    MOV AX,XYEOR            ; XX XOR YYの値を表示
    MOV SI,OFFSET MSGEOR
    CALL DSPRSLT
    MOV AX,XYADD            ; XX + YYの値を表示
    MOV SI,OFFSET MSGADD
    CALL DSPRSLT
    MOV AX,XYSUB            ; XX - YYの値を表示
    MOV SI,OFFSET MSGSUB
    CALL DSPRSLT
    MOV AX,XYMUL            ; XX * YYの値を表示
    MOV SI,OFFSET MSGMUL
    CALL DSPRSLT
    MOV AX,XYDIV            ; XX / YYの値を表示
    MOV SI,OFFSET MSGDIV
    CALL DSPRSLT
    MOV AX,XYREM            ; XX ¥ YYの値を表示
    MOV SI,OFFSET MSGREM
    CALL DSPRSLT
    .EXIT 0
EXMAC ENDP
;
; メッセージに続いて16ビットの整数値を
; 10進数、16進数、2進数で表示
;   メッセージへのポインタ:(DS:SI)
;   表示する数値:AX
;
DSPRSLT PROC NEAR PRIVATE
    PUSH AX
    INVOKE COSTR            ; メッセージを表示
    POP AX
    PUSH AX
    XOR DX,DX
    MOV BX,10           ; 10進数でAXの値を表示
    MOV CX,5
    MOV DI,OFFSET OUTBUF
    INVOKE LTOA
    MOV SI,OFFSET OUTBUF
    INVOKE COSTR
    MOV SI,OFFSET MSGCMA
    INVOKE COSTR
    POP AX
    PUSH AX
    XOR DX,DX
    MOV BX,16           ; 16進数でAXの値を表示
    MOV CX,4
    MOV DI,OFFSET OUTBUF
    INVOKE ULTOA
    MOV SI,OFFSET OUTBUF
    INVOKE COSTR
    MOV SI,OFFSET MSGCMA
    INVOKE COSTR
    POP AX
    XOR DX,DX
    MOV BX,2            ; 2進数でAXの値を表示
    MOV CX,16
    MOV DI,OFFSET OUTBUF
    INVOKE ULTOA
    MOV SI,OFFSET OUTBUF
    INVOKE COSTR
    MOV SI,OFFSET CRLF
    INVOKE COSTR
    RET
DSPRSLT ENDP
;
.DATA
XX  WORD    240
YY  WORD    35
.CONST
MSGX    BYTE    '     X = ',0
MSGY    BYTE    '     Y = ',0
MSGAND  BYTE    'X & Y = ',0
MSGIOR  BYTE    'X | Y = ',0
MSGEOR  BYTE    'X ^ Y = ',0
MSGADD  BYTE    'X + Y = ',0
MSGSUB  BYTE    'X - Y = ',0
MSGMUL  BYTE    'X * Y = ',0
MSGDIV  BYTE    'X / Y = ',0
MSGREM  BYTE    'X ¥ Y = ',0
MSGCMA  BYTE    ', ',0
CRLF    BYTE    0DH,0AH,0
.DATA?
XYAND   WORD    ?
XYIOR   WORD    ?
XYEOR   WORD    ?
XYADD   WORD    ?
XYSUB   WORD    ?
XYMUL   WORD    ?
XYDIV   WORD    ?
XYREM   WORD    ?
OUTBUF  BYTE    128 DUP(?)
.STACK
;
END
```


メモリ用語集
リアルモード　8086と同様にセグメントとオフセットでメモリにアクセスする。最大1MBのメモリ空間しかアクセスできないのも8086と同じ。

表示します．このプログラムの中で使用されているインクルードファイルと外部手続きはリスト2と同じものです．

流れ制御に関する命令とディレクティブ

流れ制御の命令とは，JMP，Jcc(ccは条件コード)，CALLなどのプログラムカウンタを変更する命令です．これらの命令の中でショートジャンプしかできないJccなどの命令は，範囲外のジャンプをしようとすると次のようなコーディングが必要でわずらわしいものです．ただし，JnccはJccの条件を否定した条件ジャンプ命令です．

```
    Jncc @F
    JMP FARAWAY
@@:
```

MASM6.0ではこのような場合，オペランドにSHORTを指定しない限り，単にJcc命令を記述するだけで上の例のコードのように展開してくれます．

流れ制御に関するディレクティブは，MASMの今回のバージョンアップでも注目に値するものです．これらのディレクティブは，構造化プログラミングの技法のうち，選択と繰り返しのコードを生成するディレクティブです．

選択を行うコードを生成するディレクティブの構文は次のとおりです．

```
.IF 実行時条件式
..........
[.ELSEIF
..........
..........]
[.ELSE
..........
..........]
.ENDIF
```

繰り返しのディレクティブには前判定型と後判定型が用意されています．前判定型の繰り返しコードを生成するディレクティブの構文は，次のとおりです．

```
.WHILE 実行時条件式
..........
..........
.ENDW
```

また，後判定型の繰り返しコードを生成するディレクティブの構文は，

```
.REPEAT
..........
..........
.UNTIL 実行時条件式
```

あるいは，

```
.REPEAT
..........
..........
.UNTILCXZ
```

です．

繰り返しループの中では，ループからの抜け出しコードを生成する．BREAKディレクティブとループの継続コードを生成する.CONTINUEディレクティブが使用できます．このディレクティブは，C言語のbreak文とcontinue文に類似する機能です．それぞれの構文は，

```
.BREAK [.IF 実行時条件式]
.CONTINUE [.IF 実行時条件式]
```

で，.IFディレクティブの付いたものは，それぞれ実行時条件が成立すると抜け出したり，継続するコードを生成します．

実行時条件式には，C言語の条件式の中で使用される比較演算子と同じ意味を持つ"==", "!=", ">", ">=", "<"，および"<="と，C言語の論理演算子と同じ意味を持つ"!", "&&"，および"||"が使用できます．これらの演算子のオペランドは，直接CMP命令やOR命令などで使用できるレジスタ式やメモリ式および定数式，あるいはZERO?, SIGN?, OVERFLOW?およびPARITY?のフラグを表すものでなければならないという制限があります．

これらのディレクティブの使用例は，掲載した各プログラムにありますので，それらを参照してください．ジャンプ関係のラベルが完全になくなって，

図5

```
name PROTO      [距離][言語][,[パラメータ]:タグ],...
     EXTERNDEF  [言語]name:修飾型
name PROC       [距離][言語][可視属性][プロローグマクロへの引数]
                [USES レジスタリスト][,パラメータ[:タグ]],...
     INVOKE     手続きを参照する式[,引数の式],...
name TYPEDEF    PROTO [距離][言語][,[パラメータ]:タグ],...
```

ソースコードがすっきりしていることがおわかりになると思います．

グローバルな資源の管理

グローバルな資源とは，多数のモジュールを開発するプロジェクトにおいて，複数のプログラムから共用される手続き（関数）や，データなどです．C言語による開発プロジェクトでは，ヘッダファイル（.Hファイル）である程度この管理を行うことができます．これとほぼ同様の機能がMASM6.0にも導入されています．

MASM6.0では，このような機能を実現するため，PROTO（プロトタイプ宣言），EXTERNDEF(外部記号の定義あるいは宣言)，INVOKE（手続きや関数の呼び出し）などの新しいディレクティブが用意され，PROC（手続きや関数の定義）ディレクティブの機能が強化されています．「データ型」の節で述べたTYPEDEFディレクティブと修飾型の導入もこの機能に欠かせないものです．図5にこれらのディレクティブに関する書式をあげます．

PROTOディレクティブは，C言語のプロトタイプ宣言のように手続きへの距離，言語規約，タグによるパラメータの数と型を宣言します．ただし，戻値の型は宣言できません．MASM6.0は，PROTOで宣言された手続きを定義するPROCディレクティブが現れると，距離，言語，パラメータの型に矛盾がないかを調べます．また，PROTOで宣言された手続きを呼び出すINVOKEディレクティブが現れると，引数に矛盾がないかどうかを調べ，可能であれば型変換を行い，宣言された言語規約と距離に従ってパラメータを手続きに引き渡すコード，手続きを呼び出すコード，およびスタックの後始末を行うコードを生成します．このように，PROTOディレクティブを使用するとグローバルな資源を定義する側と，参照する側の管理が同時に行えます．

EXTERNDEFは，従来のEXTRNとPUBLIC両者のディレクティブの機能を兼ね備えたディレクティブです．このディレクティブは，モジュールの中でnameが参照されれば外部記号として，モジュールの中でnameが定義されればパブリックシンボルとして，nameをオブジェクトファイルに出力しますが，参照も定義もされなければ何も効果をもちません．このディレクティブは，EXTRNやPUBLICのディレクティブでは管理が困難であった問題点を解消します．なお，PROTOディレクティブを使用すると，EXTERNDEFの効果も同時にもちます．

MASM6.0では，プロトタイプ宣言などのグローバルな管理情報を，INCLUDEディレクティブによってインクルードファイル（.INCファイル）から読み込みます．このインクルードファイルをC言語のヘッダファイルから作成するユーティリティがH2INCです．リスト5は，C言語のヘッダファイルであるリスト4からこのユーティリティを使用して作成したインクルードファイルです．ただし，無駄な空白や行は取り除いてあります．このときのH2INCに対して与えたコマンドラインは次のとおりです．

```
H2INC /Fa EXGRC.H
```

このインクルードファイルを利用して，簡単なグラフィックのプログラムを作成したのが，リスト6です．1行目にEXGRC.INCを読み込むよう指示されています．INCLUDELIBディレクティブで指定されているGRAPH98.LIBは，筆者がMS-C6.0用に開発したPC-98シリーズ用

リスト4　EXGRC.H

```
void _far gcls(void);
int _far getres(void);
void _far ginit(void);
void _far gline(int x0, int y0, int x1, int y1, int pltn);
void _far gscreen(int mode, int sw, int actp, int disp);
void _far gterm(void);
```

リスト5　EXGRC.INC

```
@proto_0 TYPEDEF PROTO FAR C
gcls PROTO @proto_0
@proto_1 TYPEDEF PROTO FAR C
getres PROTO @proto_1
@proto_2 TYPEDEF PROTO FAR C
ginit PROTO @proto_2
@proto_3 TYPEDEF PROTO FAR C :SWORD, :SWORD, :SWORD, :SWORD, :SWORD
gline PROTO @proto_3
@proto_4 TYPEDEF PROTO FAR C :SWORD, :SWORD, :SWORD, :SWORD
gscreen PROTO @proto_4
@proto_5 TYPEDEF PROTO FAR C
gterm PROTO @proto_5
```


メモリ用語集
ローカルアドレス　プロテクトモードでは，ほかのタスクとのメモリのバッティングやOSの保護を目的としてグローバルアドレス（絶対アドレス）を参照することはできない．

リスト6 EXRG.ASM

```
INCLUDE EXGRC.INC
INCLUDE EXSUB.INC
INCLUDELIB GRAPH98
;
.DATA?
NDAT    WORD    ?   ; 座標データの個数
STEP    WORD    ?   ; 座標データのバイト間隔
PL  WORD    ?   ; パレット番号
X0  SWORD   ?   ; 始点のX座標
Y0  SWORD   ?   ; 始点のY座標
X1  SWORD   ?   ; 終点のX座標
Y1  SWORD   ?   ; 終点のY座標
;
.CODE
;
; 与えられた座標をすべて直線で結ぶ
;
EXGRAPH PROC
     .STARTUP
     MOV AX,DS
     MOV ES,AX
     MOV SI,OFFSET CLSDAT     ; テキスト画面消去
     INVOKE COSTR
     INVOKE ginit          ; グラフィック関数の初期化
     INVOKE getres         ; 解像度の種別を取得
     .IF AX!=0   ; ハイレゾリューションの場合
         INVOKE gscreen, 0, 0, 0, 1
         MOV WORD PTR PL,15  ; パレット番号15:=白色
         MOV WORD PTR NDAT,23     ; 座標データ数:=24点
         MOV WORD PTR STEP,4 ; 座標データの間隔
         MOV SI,OFFSET HIRES ; 座標データの先頭
     .ELSE        ; ノーマルレゾリューションの場合
         INVOKE gscreen, 3, 0, 0, 1
         MOV WORD PTR PL,7   ; パレット番号7:=白色
         MOV WORD PTR NDAT,11     ; 座標データ数:=11点
         MOV WORD PTR STEP,8 ; 座標データの間隔
         MOV SI,OFFSET LORES ; 座標データの先頭
     .ENDIF
     MOV CX,NDAT
     .REPEAT
         MOV AX,[SI] ; 始点の座標を求める
         MOV X0,AX
         MOV AX,2[SI]
         MOV Y0,AX
         MOV DI,SI
         ADD DI,STEP
         PUSH CX
         .REPEAT
             MOV AX,[DI] ; 終点の座標を求める
             MOV X1,AX
             MOV AX,2[DI]
             MOV Y1,AX
             PUSH CX
             INVOKE gline, X0, Y0, X1, Y1, PL
             POP CX
             ADD DI,STEP
         .UNTILCXZ
         POP CX
         ADD SI,STEP
     .UNTILCXZ
     INVOKE gterm    ; 後始末
     .EXIT 0
EXGRAPH ENDP
;
.CONST
; ノーマルレゾリューション用データ
LORES   WORD     499,199,493,246,475,289,446,326
     WORD    409,355,366,373,319,379,272,373
     WORD    229,355,192,326,163,289,145,246
     WORD    139,199,145,152,163,109,192,72
     WORD    229,43,272,25,319,19,366,25
     WORD    409,43,446,72,475,109,493,152
; ハイレゾリューション用データ
HIRES   WORD     955,365,942,458,906,545,849,620
     WORD    775,677,688,713,595,725,501,713
     WORD    415,677,340,620,283,545,247,458
     WORD    235,365,247,272,283,185,340,110
     WORD    415,53,501,17,594,5,688,17
     WORD    774,53,849,110,906,185,942,272
; テキスト画面消去のエスケープシーケンス
CLSDAT  BYTE    1BH,'[2J',0
;
.STACK 2048
;
END
```

のグラフィックライブラリです．このプログラムは，最初にテキスト画面を消去し，グラフィック関数使用のための初期化を行った後，ハードウェアのレゾリューションを識別します．次に，レゾリューションに応じた画面の初期化を行い，あらかじめ計算されている円周上の座標を直線によって総当たりでつなぎます．ノーマルレゾリューションでは12個，ハイレゾリューションでは24個の頂点を結んでいます．

MASM6.0 あと書き

MASM6.0については，あまりに紹介しなければならないことが多かったため，かなり内容をはしょってしまいました．MASM5.1との互換性についても触れたかったのですが，ついに紹介できませんでした．ただ，筆者の手元にあるソース約100本をMASM6.0のデフォルトの状態で片っ端からアセンブルしてみたのですが，9割程度は問題ありませんでした．問題で多かったのは，予約語とラベルの衝突でした．そのほかでは，-2[SI]のような式と生成される命令コードに依存したコードに関する問題が少数発生しました．

MASM6.0を使用した雑感としていえるのは，かなり文法が複雑怪奇になってきているということです．ディレクティブによって使える式の種別が異なったり，同じ演算子が異なる意味に使用されたりで，初心者にとってはかなり難解でしょう．これは以前のバージョンとの互換性を重視してきたことも一因でしょうが，このへんで言語の一貫性を見なおすべきときではないかと思われます．それでも，以上紹介した機能，とくに構造化プログラミングの導入などは十分に試してみる価値があります．

最後になりましたが，例にあげたプログラムをアセンブルするためのバッチプログラムEXMAK.BAT，各プログラムが共通に使っている手続きのソースプログラムEXSUB.ASMとインクルードファイルEXSUB.INCなどは，付録のフロッピーディスクに格納されているはずです．興味のある方はご一読ください．

メモリ用語集
論理ページ　拡張メモリのセグメント．EMSメモリの論理ページは通常16KBである．各論理ページにアクセスするにはマッピングする必要がある．

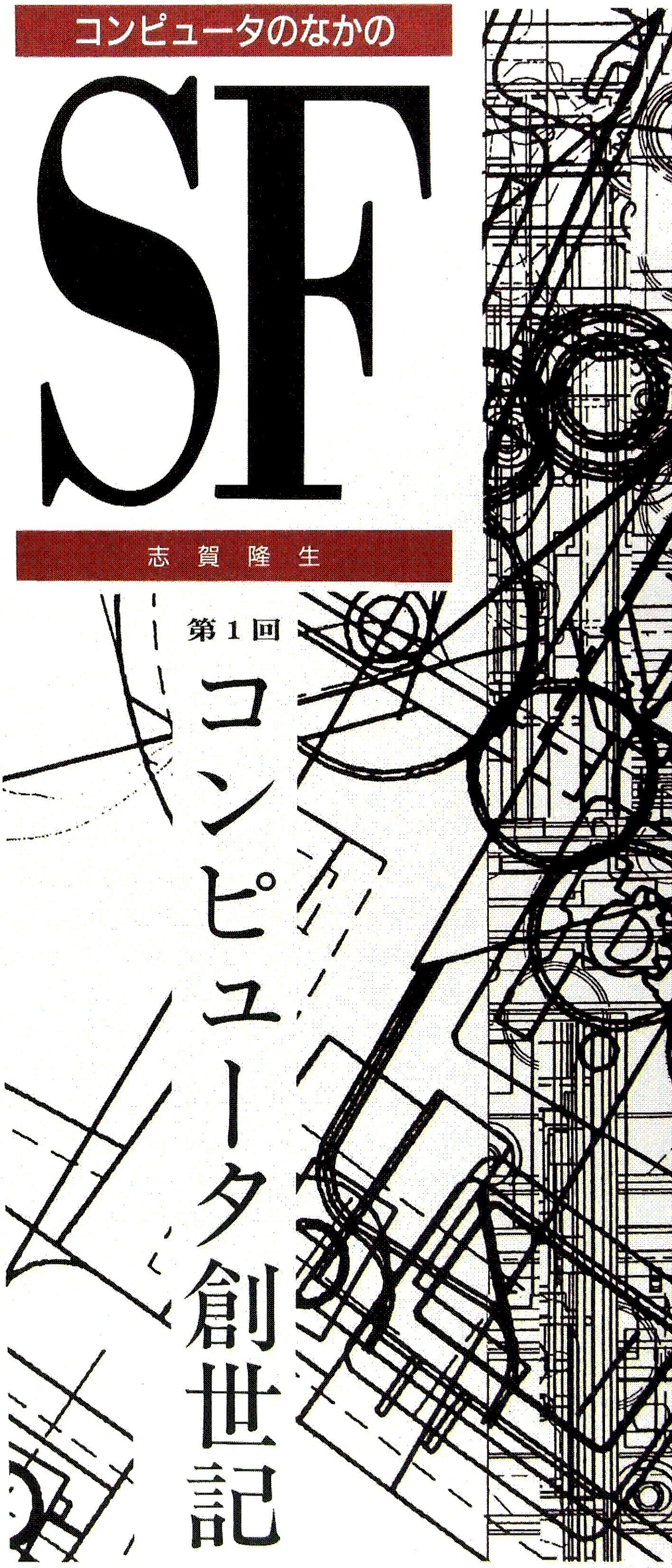
コンピュータのなかの

SF

志 賀 隆 生

第1回

コンピュータ創世記



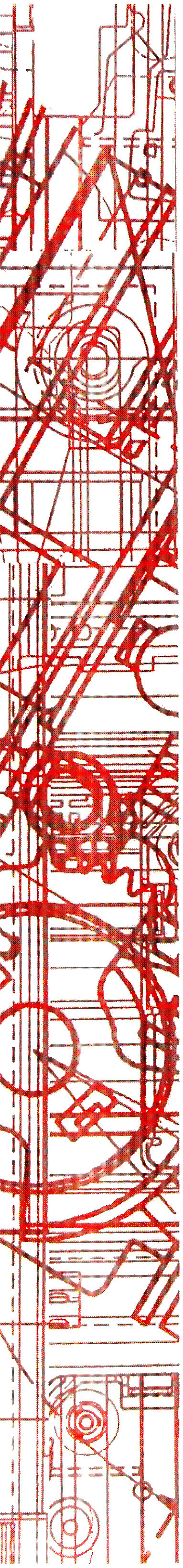
1992年1月12日．この日が何の日だかわかりますか？　そう，映画『2001年宇宙の旅』に登場するHAL9000の誕生日．映画では後半のクライマックスの直前のシーン，決死の緊急入船を果たした後，HALの活動を止めるため，ボーマン船長がメモリユニットを1枚1枚引き抜いていく．すると――．

「皆さん，こんにちわ．私は，ハル9000コンピュータです．イリノイのHAL工場で生まれました．1992年1月12日でした．教師はラングレーさん．歌うことも教わりました．お望みなら，お聞かせします．題は「デイジー」です．“デイジー…デイジー…答えておくれ，僕は夢中…”」

薄れゆく意識（？）の中で，HALの過去の記憶があふれだす．コンピュータエンジニアならずとも，胸を打つ場面だ．映画には，HALに代表されるような，十分に科学考証を加えられたアイテムが数多く登場する．NASAの技術者をはじめとして，当時の第一線の科学者，技術者が呼び集められ，アイデアを出し合って，『2001年宇宙の旅』は完成したのだった．映画として優れていたのはもちろん，未来予測映画としても，十分に機能を果たすものだった．

しかし，その途方もない作業に原作者として参加したA・C・クラークは，後にこう述懐している．多くの予測は満足のいく成果を上げたが，コンピュータの進歩は読み誤っていた，と．さらに，現在の技術の水準を見れば，HALの中央制御室のスペースはもっと小さくてすんだだろう，とも語っている．クラークはまた別のところで，多くのSF作家は未来予測に心血を注いできたが，60年代半ばまでのSFはコンピュータの描写場面が出ると途端に色褪せてしまう，とも書いている．なぜなら，半導体の進歩がこれほど急速だとは誰も思わなかったから．超小型真空管がずらりと並んだ灼熱の中央演算室というのは，レトロ風味はあっても，シリアスな雰囲気は出てこないだろう．

クラークならずとも，コンピュータの進歩のスピードには驚かないではいられない．コ



ンピュータの進歩は他のテクノロジーの大幅な進歩も促して，今日ではSF的未来予測（かつて言われた〈これがもしこのまま続けば〉というエクストラポレーション）の魅力をほとんど無効なものにしている．高温超伝導や常温核融合（事の当否はともかく），あるいは急速に現実味を帯びてきたナノテクノロジーをみればわかるとおり，進歩は連続的に起こるのではなく，階段上に，非連続的に進んでいる．では，今日のSFの役割は何かといえば，巷間にあふれる未来の変化の予兆を私たちのイマジネーションがどのように感じ取っているのか，それをフィクションという装置を通してトレースしてみせることだろう．

というわけで，本欄では，SF作品に登場するコンピュータを紹介しながら，〈現在〉の中に入り込んでいる多様な〈未来〉を，その変化の予兆を感じ取ってみたいと思う．まず初めに取り上げるのは，コンピュータ誕生の物語だ．

物事にはなんでも始まりがある．始まりを知ることは，そのおおもとの本質を知るうえで何らかの役に立つことが多い．たとえば，コンピュータの発達史をチューリングやノイマン，ウィナーらの仕事をたどりながら考察するハワード・ラインゴールドの好著『思考のための道具』では，19世紀の数学者で差分機関（ディファレンス・エンジン），解析機関（アナリシス・エンジン）の発明者，風変わりな天才チャールズ・バベッジと，そのよき理解者だったバイロンの娘（競馬狂でもあった）エイダ・ラブレイス伯爵夫人のエピソードから始まっている．バベッジは機械式であるとはいえ，今日のコンピュータの考えをほぼ先取りするアイデアを提出し，相棒のエイダは今日のプログラム方式のコンピュータを先取りする発想を持っていた．

バベッジは全財産を投げ打って努力したが，当時の工作技術の限界から，ついに解析機関を完成させることはできなかった．そこで，もし，バベッジらが活躍した19世紀半ばに，機械式コンピュータが完成したとしたら，当時の世界はどのように変化しただろうか――．ウィリアム・ギブスンとブルース・スターリングの共著『ディファレンス・エンジン』はまさに，そうしたIFの世界を主題にしたSFである．

「そこは，人々が蒸気タイプライターで執筆し，蒸気映像を観賞し，蒸気自動車を乗り回し，蒸気クレジット・サービスを活用し，はては蒸気コンピュータ・ハッカー（クラッカー）作成するところの蒸気コンピュータ・ウイルスさえ跳梁する幻想の大英帝国．」（同書解説より）

本書の舞台となるのは一九世紀半ば，産業革命が一段落したものの，まだ社会全体が騒然としていた時代である．世の中にモノはあふれているが，階級制度は厳然として存在し，労働者は抑圧と貧困にあえいでいた．一方で，産業革命で手にした科学力，軍事力，資本力によって，イギリスは世界の覇者への道を着々と進んでいた．新時代の希望と旧時代の体制の崩壊が同居し，まさに光と闇が激しく交錯する不思議な時間帯．これなら何が起きても不思議ではない．

ギブスンとスターリングはサイバーパンクで一世を風靡した気鋭のアメリカSF作家だが，ここでは近未来のハイテク都市から一転して，光と闇が激しく交錯する歴史のなかの時代と都市を選んだ．そこで描かれているのは，蒸気コンピュータの出現によって激しく変貌していく〈過去〉なのだが，しかし，それは拡大された〈現在〉でもある．コンピュータは未来へと向かう〈夢の技術〉のように一般にはとらえられているが，ここでは，コンピュータは19世紀という過去＝機械の時代の〈機械の亡霊〉のようにも見える．本書を読み通して，ふと周囲を見渡すと，最新のハイテク機器が，何かしら古びて見えるから不思議である．この辺が本書の最大の読み所だろう．

●『思考のための道具』（Tools For Thought, 1985）H・ラインゴールド，栗田昭平監訳，パーソナルメディア

●『ディファレンス・エンジン』（The Difference Engine, 1990）W・ギブスン，B・スターリング，黒丸尚訳，角川書店

祐安重夫

プログラマの道具学

第16回　胃腸薬

ソフトウェアの仕事というのは外からみているのとはちがって，頭脳労働というよりは肉体労働にちかい部分がおおい．納期も間近になると，ほとんど体力だけが勝負の世界になってしまい，思考が完全に停止した状態で徹夜をつづけたりもする．

徹夜がつづいて肉体的にも精神的にもまいっている時は，胃の具合もおかしくなって食欲などなくなっている．こういう時はおとなしく胃腸薬でものんでおけばいいのだが，つい酒をのんでしまい，それで二日酔になってからあらためて，胃腸薬のお世話になったりする．

もちろん1日中机やキーボードにむかっていると，結構精神的なストレスもたまってくる．夜おそくまで仕事をしていると，神経がたかぶってねむれなくなり，その結果としてつい酒の量がすぎてしまうということも，私に関していえばめずらしくはない．またCRTのみつめすぎで肩がこってしまい，そのいたみをわすれるために酒をのみ，さらに肩こりを悪化させることもある．

さらにこのように文章をかく仕事もやっていると，酒の異常につよい編集者諸兄（たとえば本誌の版元の社長や編集者など）とのおつきあいもふえて，またまた酒がすぎてしまうことになる．うちあわせと称してはのみにでかけ，原稿ができたといってはうちあげをし，本がでれば出版記念会といった具合で，こちらもけっしてきらいなほうではないので，さそいをことわることはほとんどない．おかげで翌朝は二日酔となり，胃腸薬のお世話になることも，しばしばある．

もちろん原稿も，最近ではキーボードにむかって執筆していくことになるので，脱稿後の深酒というパターンも当然のごとくにある．もっとも原稿の方は，ソフトウェアほどはストレスの原因にならないようだが．

Illustration
Mikio Sasaki

その昔連載をしていた某誌の編集長（一応友人でもあった）など，原稿料が振り込まれた直後になるとかならず電話をかけてきて，深夜まで勘定はこちらもちで，下手なカラオケに毎回つきあわされたものだったが，この時は翌朝かならずひどい二日酔になやまされた．もっともこの人物は，その後とある酒の席によばなかったことを根にもって，不気味な絶交状をよこしたので，最近ではここまでわるい酒につきあう機会は，あまりなくなった．

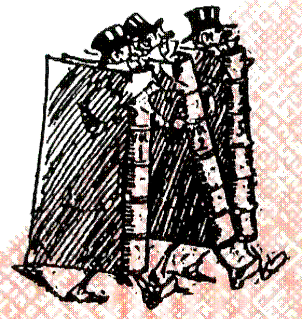
高校時代からの友人の名言に，「たべすぎには三共胃腸薬，のみすぎには太田胃散」というのがあるのだが，私も太田胃散がてばなせないでいるのは，じつにこまったものである．もっとも最近ではソルマックなどの二日酔専門の薬もでているようで，これもときどきはためしてみている．

二日酔の日は太田胃散にむせながら，当分酒はやめようなどと殊勝なことをかんがえているのだが，夕方になると元気になってきて，結局アルコールをのまない日は，ほとんどないというありさまである．

ソフトウェアの仕事は体によくないというべきなのか，それともこれは，単に私の個人的な事情にすぎないのか，もうひとつよくわからないところがある．

IC千夜一夜

第10回——ミニコンピュータのマイクロコンピュータ化



虎岩 登

ミニコンピュータの登場 大型化の潮流に逆らって

今月は，ミニコンピュータのマイクロコンピュータ化について見ていきます．

以前からコンピュータ業界にいる方を除けば，いまではミニコンピュータとか，ミニコンという言葉を知らない方も多くなっているだろうと思います．ミニコンピュータとは，1960年代から70年代にかけて登場したその名の通り小型のコンピュータで，数百万円から数千万円も出せばシステム全体を購入することのできた小規模なものを指します．一般にコンピュータは大型化すればするほどコストパフォーマンスがよくなると信じられていたため，世の中ではコンピュータの大型化が進んでいたときに，その傾向に反して小回りを活かしたコンピュータとして，制御用・(予算の少ない部門の)研究用・教育用などに使われていました．そうはいっても，いま考えると十分に高価で図体の大きいものだと思いますが．

1970年前後のミニコンピュータは，主要論理回路は比較的小規模なIC，多くはTTL-ICを大量に用いています．CPUというと，パーソナル・コンピュータばかり利用している方はLSI 1個を思い浮かべるかもしれませんが，ミニコンピュータのCPUはIC数百個を50cm四方もあるような基板数枚に実装したものでした．CPU部分の性能は，そうですね，いまのノートブック・タイプのコンピュータと同程度かそれ以下だったでしょう．メイン・メモリ容量だけなら完全に負けてしまいます．もちろん，ミニコンピュータ自体も性能の向上はありましたし，高価なモデルはそれなりの性能ではありました．それに，いまどきのパーソナルコンピュータよりは多くの周辺機器や端末をサポートするようになっていました．しかし，いまでは，スーパー・ミニコンピュータという系列も，ワーク・ステーションの発展によって影が薄くなってきています．

それでも，ある時代には「ミニコンピュータ」は十分に存在感のあるグループを構成していました．

ミニコンピュータ そのLSI化の流れ

1970年代前半には，インテルとモトローラから8bitマイクロプロセッサが発表されます．そして，下位ミニコン

ピュータと同程度の用途，あるいは下位ミニコンピュータではコスト的に厳しかった分野などにマイクロコンピュータが進出していきます．それでは，このときミニコンピュータのメーカーはどのような反応をしたのでしょうか．

出足は遅れたものの，やはり，ほとんどすべてのメーカーがLSI化したミニコンピュータを出してきました．したがって，LSI化されたミニコンピュータは，マイクロコンピュータ化したミニコンピュータと呼ばれるべき代物です．元来ミニコンピュータは簡単なハードウェアで低コストのコンピュータなのですから，そのCPUボードをLSIに押し込めることは比較的容易なことでした．

ミニコンピュータ界の王者，DEC社は1975年2月にLSI-11マイクロコンピュータを発表しました．これは同社の主力ミニコンピュータPDP-11をLSI化したモデルです．構成は図1を参照してください．主要LSIはWestern Digital（ウエスタン・ディジタル）社のMCP-1600チップセットを採用しています．機能的には3種類のLSIがマイクロプログラム制御で，機械命令を実行しています．1つはシーケンサなど全体の制御を行うコントロール・チップ，そして算術論理演算ユニット（ALU）やレジスタ・ファイルを含むデータ・チップ，最後にそれらのチップにマイクロ命令を供給する22bit×512ワード構成のマイクロム（MICROM）チップです．さらに，コントロール・チップとデータ・チップは1つずつ，マイクロム・チップは標準で2個でCPUを構成しています．マイクロム・チップは1個をオプションで実装することにより，乗除算命令と浮動小数点命令を利用できるようになっています．マイクロ命令の実行サイクルは380ns，命令実行時間は，レジスターレジスタ間のMOV命令で3.8μsといったところでしょうか．

8KByteのメモリと一緒に，21.6×25.4cmの基板に16bit CPUのすべてが含まれています．ちょうど本誌の1ページよりちょっと大きいくらいのサ

図1　LSI-11の構成

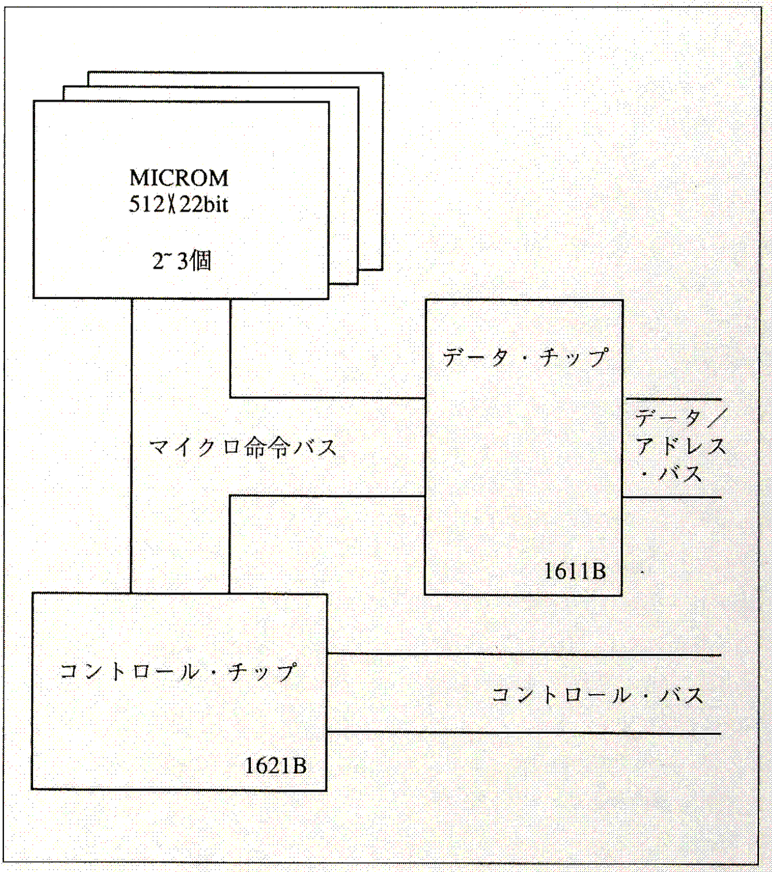
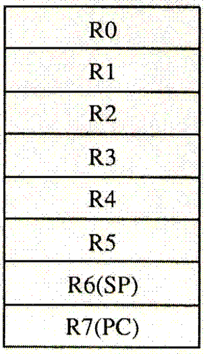


図2　LSI-11のレジスタ・セット



LSI-11のレジスタ・セット
R6はスタックポインタ，R7はプログラムカウンタを兼用する．

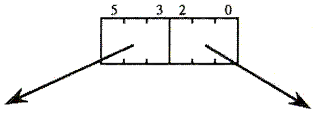
イズです．もちろん，これはCPUだけの基板ですから，使用する上ではRS-232Cなどの通信インターフェースやハードディスクインターフェースなど数枚の基板を加えなければなりません．それでも，ハードディスクや端末を除く部分が，いまのパーソナル・コンピュータと同じくらいの筐体に納まったのですから十分に画期的だと思います．

このLSI-11は，同社の中級ミニコンピュータであったPDP-11/34と命令コンパチブルになっていて，膨大なソフトウェア資産を継承できるようになっています．ミニコンピュータの代表として，レジスタ・セット（図2）とアドレッシング・モード（図3），命令の基本フォーマット（図4）を示しておきます．

さらに1977年11月には，LSI-11からオンボードのメモリを除いたLSI-11/2を発表します．このLSI-11/2のサイズは，名前のとおり，LSI-11のサイズの1/2になっています．すなわち，21.6×12.7cmです．命令セットなどは完全に互換性があります．LSIは同様の構成です．

ほかには，昔からDECと並んで人気の高かったミニコンピュータ，データ・ジェネラル社のNOVAのワンチップLSI化が1976年に同社によって行われています．LSIの名前はmN601，コンピュータシステムとしての商品名は，micro NOVAです．半導体メーカーのフェアチャイルド社は，NOVAと命令が完全コンパチブルなマイクロプロセッサ，9440 MICROFLAMEを出しています．こちらはバイポーラ・プロセスであるI^3Lを用いためずらしいマイクロプロセッサです．また，インターシル社は，DEC社のPDP-8Eという12bitミニコンピュータをマイク

図3　LSI-11のアドレッシング・モード



アドレッシング・モード指定ビット

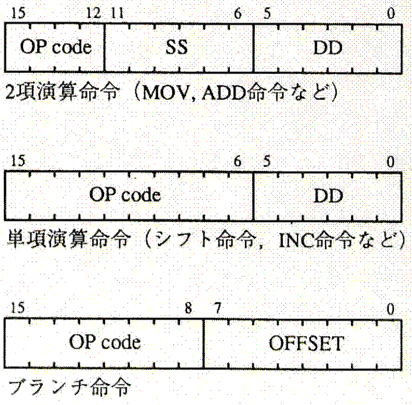
0	レジスタ直接
1	レジスタ間接
2	レジスタ・オートインクリメント R7の時，イミディエート
3	レジスタ間接オートインクリメント R7の時，絶対番地
4	レジスタ・オートデクリメント
5	レジスタ間接オートデクリメント
6	インデックス R7の時，PC相対
7	インデックス間接 R7の時，PC相対間接

レジスタ指定ビット

0	R0
1	R1
2	R2
3	R3
4	R4
5	R5
6	R6(SP)
7	R7(PC)

アドレッシング・モードの指定方法．レジスタ指定ビットがR7(PC)を示すときには特別の意味になる．

図4　LSI-11の命令の基本フォーマット



LSI-11の代表的命令フォーマット．
SS, DDはそれぞれソース・オペランド，
ディスティネーション・オペランドのアドレッシングモード

ロコンピュータ化して，IM6100シリーズとして供給していました．

日本では，東芝が自社のミニコンピュータTOSBAC-40をLSI化したTOSBAC-40L(T40L)を発表しています．これも，マイクロプログラム制御で機械命令を実行する構成ですが，マイクロプログラムを除く主要部分をT3412というLSI 1個にまとめたものです．

マイクロコンピュータ化 その意義と影響

当時，マイクロコンピュータは生まれたばかりで，たいしたソフトウェア資産をもっていませんでした．マイクロコンピュータ・システム上でそのソフトウェアを開発する，セルフ開発環境などほとんどなく，マイクロコンピュータのソフトウェア開発にはミニコンピュータを使うクロス開発を用いるのが普通だったくらいです．当然，ミニコンピュータはマイクロコンピュータとは比較にならないくらい多量のソフトウェアを所有していました．半導体メーカーが，ミニコンピュータと命令体系の同じCPUを開発したのも，そのソフトウェア資産を活かすためだったのです．4bitないし8bitのマイクロコンピュータの応用では，貧弱なソフトウェア開発システムでも開発できるような，小規模な応用が多いので何とかなるだろうが，16bit以上のマイクロコンピュータを用いる大きな応用では本格的な開発用ソフトウェアが必要であろうと考えられるためです．

それなりのオペレーティング・システムはもちろん，その上で動作する高級言語も多数ありました．ミニコンピュータは，対話的に使用することも多かったため，ちょうどいまのパーソナルコンピュータ上で開発を行うのと似たような感じでプログラミングが可能でした．組み込み用途に用いるため，セルフ開発が可能なほどにはメモリや周辺機器が接続されていない場合でも，同じ命令を実行可能なミニコンピュータで開発できますから，MS-DOS上で8086のワンボード・コンピュータのプログラム開発を行うのと同じように，比較的容易に開発が進められます．

マイクロコンピュータ化されたことで，ミニコンはたしかに小型で安価にはなりました．性能的にも，パーソナル・コンピュータとして用いるには，当時の水準からすれば十分すぎるほどありました．なにしろ，大量のソフトウェア資産がある16bitコンピュータですから．パーソナル・ユースには，ヒースキットから美しい筐体に納められたLSI-11/2も発売されていました．

しかし，パーソナル・コンピュータ以下の市場には，ミニコンピュータはほとんど現れませんでした．1つには，ハードウェアが高価であったためです．CPU単体やメモリは徐々に値下がり傾向がありましたが，フロッピーディスクやハードディスクなどの外部記憶装置，CRTターミナルやプリンタなどの端末が，まだ非常に高価でした．フロッピーディスク2台接続するだけで100万円前後かかるとなれば，システムとしては高価になりすぎて，導入は困難です．せっかくミニコンピュータを導入しても，紙テープにテレタイプライタではたいした仕事もできません．

現在のパーソナル・コンピュータとその市場が成立するには，マイクロコンピュータによって安価なCPUが広まるにしたがって，その後を追いかけて周辺機器が安価に，高性能になってきたという理由を忘れてはなりません．もっとも，その周辺機器が高性能になりながら安価になっていったのも，市場の広がりにつれて量産化されるようになったことのほかに，周辺機器の制御部がLSI化され安価なマイクロコンピュータが応用されるようになってきたのも一因です．

その後，DEC社も，LSI-11/2と同サイズでメモリやI/Oを備えたFALCON SBC-11/21や，CPU自体の能力を向上したLSI-11/23やLSI-11/73を発表していきますが，マイクロコンピュータの高機能化にしたがってこの高名な下位ミニコンピュータ・シリーズも消えていきます．同社は主力を32bitスーパー・ミニコンピュータVAXシリーズに移していきました．その後は，ワークステーションとの競争などありますが，ミニコンピュータの時代に貯えた資産はたいしたものだと思います．

そういえば，現在のMS-DOSは，PDP-11のOS，RT-11やRSX-11の影響を受けています．MS-DOSが手本にしたCP/MというOSは，RT-11などを手本に簡略化したものです．また，MC68000など多くの16bitマイクロコンピュータは，ミニコンピュータの命令体系を参考にしたところがみられます．なんとなく，MC68000のマニュアルはPDP-11のマニュアルと表記が似ています．

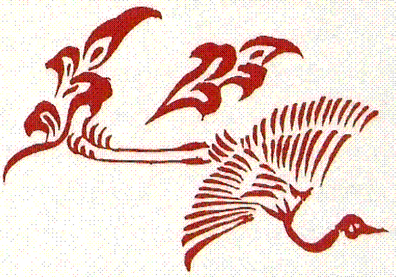
詳細には踏み込みませんが，命令体系も基本部分はシンプルで美しく，会社によってそれぞれに個性もあって，アーキテクチャの理解しやすいコンピュータでした．(下位)ミニコンピュータは，半導体技術の進歩によって淘汰されていった——あるいは発展的に解消していった——のだといえるでしょう．

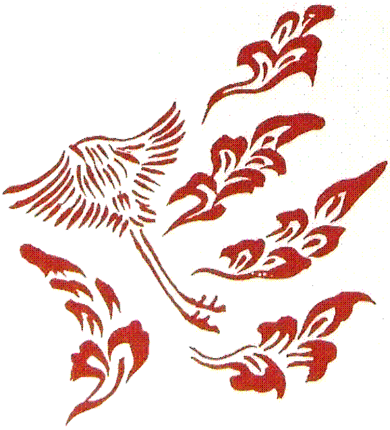
プログラム遊工房

谷口典生

拾伍乃巻

trackrw／フロッピーディスクの直接読み書き





今回は，フロッピーディスクをトラック単位で読み書きするツールを作りました．前回のバイナリエディタと合わせてディスクエディタを作ってみてはいかがでしょうか．
ワンポイントテクニックでは，DISK BIOSの利用方法を説明しています．DISK BIOSの扱いはディスクのフォーマットに関する知識が必要なのでこれだけでは理解できないかもしれませんが，なにかの参考にしてもらえればよいと思います．

リスト1

```
/*
 * trackrw.c            floppy disk track Read/Write
 */

#include <stdio.h>
#include <stdlib.h>
#include <dos.h>

#ifdef __TURBOC__
#define _asm    asm
#endif

#define BUFFLEN              10000           /* トラックバッファの長さ */
#define READ_DATA            6               /* read data */
#define READ_DELETED_DATA    12              /* read deleted data */
#define WRITE_DATA           5               /* write data */

unsigned char pda;                       /* 対象ドライブのPDA */
int dens;                                /* 記録密度(0=単密、1=倍密) */
int track;                               /* トラック番号 */
unsigned char *buffer;                   /* トラックバッファ */
int datalen, filelen;                    /* データの合計、ファイル長 */
int id_c, id_h, id_n, max_r;             /* IDのC、H、NとRの最大値 */
int writeflag=0;                         /* 書き込みフラグ */

int sense()                              /* ドライブの状態を取得 */
{
        union REGS regs;

        regs.h.ah = 0x64;                /* sense status */
        regs.h.al = pda;                 /* デバイス及びユニット番号指定 */
        int86(0x1b, &regs, &regs);       /* Disk BIOS 呼び出し */
        return (int)regs.h.ah & 0xf0;    /* リターンコード */
}

int readid(long *chrn)                   /* read ID */
{
        union REGS regs;

        regs.h.cl = track >> 1;          /* シリンダ番号 */
        regs.h.dh = track & 1;           /* ヘッド番号 */
        regs.h.al = pda;                 /* デバイス及びユニット番号指定 */
        regs.h.ah = dens==0 ? 0x3a : 0x7a;          /* read ID */
        int86(0x1b, &regs, &regs);       /* Disk BIOS 呼び出し */
        *chrn = ((long)regs.x.dx << 16) + regs.x.cx;    /* IDのC,H,R,N */
        return (int)regs.h.ah & 0xf0;    /* リターンコード */
}

int rw_data(int command)         /* データリード/ライト */
{
        _asm {
                push    es
                push    bp
                mov     ax, dens
                and     ax, 1
                mov     cl, 14
                shl     ax, cl
                mov     bx, command
                or      ah, bl                  /* AH : command */
                mov     al, pda                 /* AL : デバイス */
                push    ax
                mov     cx, id_c                /* CL : IDのC(シリンダ) */
                mov     ax, id_h
                mov     dh, al                  /* DH : IDのH(ヘッド) */
                mov     ax, id_n
```

図1

```
デバイス種別  : 2HD 倍密
トラック番号  : 0
セクタ長      : 1024(400h)bytes
セクタ数      : 8
データ長      : 8192(2000h)bytes
ファイル名    : c_000.dat

トラック内容をファイルに読み出しました.
```

プログラムの使い方

このプログラムはPC-9801シリーズ用で，Turbo C，QuickCおよびMS-Cに対応しています．リスト1を打ち込み，コンパイルして実行ファイルを作ってください．

プログラムの実行は

```
trackrw <d:> <track> [/W]
```

と，読み出したいドライブ名とトラック番号を指定して起動します．書き込みの場合はスイッチ「/W」をつけてください．

たとえばCドライブに2HDのMS-DOSフォーマットのディスクが挿入されているとき，

```
trackrw c: 0
```

を実行すると図1のように表示され，0トラックのデータが「c_000.dat」に読み出されます．このファイル名は「ドライブ名+'_'+トラック番号+'.dat'」として作られます．

2HD/2DD両用のドライブを使用していれば，自動的に2HD倍密，2HD単密，2DD倍密，2HD単密の順にセクタを探し，使用しているセクタ長とセクタ数を判定してデータを読み出します（フォーマットされていないトラックを指定すると「読めません」と表示して終了します）．

```
		mov	ch, al			/* CH : IDのN(セクタ長) */
		mov	dl, 1			/* DL : IDのR(セクタ番号) */
		pop	ax
		push	ds
		pop	es			/* ES : バッファのセグメント */
		mov	bp, buffer		/* BP : バッファのオフセット */
		mov	bx, datalen		/* BX : 処理するデータ長 */
		int	0x1b			/* Disk BIOS 呼び出し */
		mov	al, ah
		and	al, 0xf0		/* リターンコード */
		pop	bp
		pop	es
	}
}

int idcheck()			/* ID を読み出す */
{
	int stat;			/* BIOSリターンコード */
	long id;			/* ID(C,H,R,N) */
	int tmp_r, top_r;		/* 現在のR、最初に読み出したR */

	stat = readid(&id);		/* IDを読み出す */
	if(stat!=0)			/* ひとつも読み出せない */
		return stat;

	top_r = (id>>16) & 0xff;	/* 最初に読み出したIDのR(セクタ番号) */
	max_r = 0;
	do {
		stat = readid(&id);		/* 次のIDを読み出す */
		if(stat!=0)
			return stat;
		tmp_r = (id>>16) & 0xff;
		if(max_r<tmp_r)
			max_r = tmp_r;		/* 最大セクタ番号を更新 */
	} while(tmp_r!=top_r);		/* 最初と同じIDが読み出されるまで */

	id_c = id & 0xff;		/* IDのC(シリンダ番号) */
	id_h = (id>>24) & 0xff;		/* IDのH(ヘッド番号) */
	id_n = (id>>8) & 0xff;		/* IDのN(セクタ長) */
	return 0;			/* 正常終了 */
}

void drivecheck()		/* ドライブの状態をチェックする */
{
	int stat;			/* BIOSリターンコード */

	stat = sense();			/* ドライブ状態の読み出し */
	switch(stat) {
	case 0:				/* エラーなし */
		break;
	case 0x10:			/* write protect */
		if(writeflag==1) {
			fprintf(stderr, "書き込み禁止です.¥n");
			exit(1);
		}
		break;
	case 0x60:			/* drive not ready */
		fprintf(stderr, "ドライブの準備ができていません.¥n");
		exit(1);
	default:			/* その他のエラー */
		fprintf(stderr, "エラー[%x]が発生しました.¥n", stat);
		exit(1);
	}
}

void usage()			/* 使い方の表示 */
{
	printf("Usage: trackrw <d:> <track> [/W]¥n");
	printf("¥t<d:>......ドライブ名¥n");
	printf("¥t<track>...トラック番号¥n");
	printf("¥t/W........書き込み指定¥n");
}

void allocate()		/* バッファにメモリを割り当てる */
{
	long addr;	/* バッファの先頭アドレス(24bit絶対値換算) */

	buffer = malloc((size_t)BUFFLEN);
	if(buffer==NULL) {
		fprintf(stderr, "メモリが足りません.¥n");
		exit(1);
	}

	/* バッファが 64K boundary にまたがる場合、再度割り当てを行なう */

	addr = ((long)buffer & 0xffffL) + (((long)buffer>>12) & 0xffff0L);
	if((addr & 0xffffL) + BUFFLEN > 0x10000L) {
		buffer = malloc((size_t)BUFFLEN);
		if(buffer==NULL) {
			fprintf(stderr, "メモリが足りません.¥n");
			exit(1);
		}
```

「/W」スイッチが指定されていれば，同様にファイルからトラックへの書き込みを行います．ファイルの長さが指定したトラックのデータ長（セクタ長×セクタ数）と一致していなければエラー終了します．

このプログラムではIDのシリンダ番号が実際のシリンダ番号と一致していなくてもデータを読み出せるので，たとえば2Dのディスクを2DD(または2HD/2DD両用)ドライブで読み出すことができますが，2Dのディスクに書き込みを行った場合，以後2Dドライブで正常に読み出せなくなる場合があるので注意してください(2Dドライブと2DDドライブではヘッド幅が違うため)．

単密/倍密は記録方式の違い(倍密度記録方式は単密の2倍のデータが記録できる)で，MS-DOSでは倍密しか使われていませんが，DISK BASICの0トラックなど単密のセクタが使われているものも存在します．

通常使われているフォーマットはほぼ問題なく読めるはずですが，コピープロテクトのかかった特殊なトラックなどは正常に読めないことがあります．このプログラムで正常に読み書きできるトラックは，

①IDのC，H，Nはすべてのセクタで一致していること
②IDのRは1から最大値まで1つずつ存在すること（セクタの並びは任意）
③IDのHの最下位ビットと実際のヘッド番号が一致していること

といった条件を満たすもので，セクタの合計が10KBを越えたり(特殊な方法を使わない限り越えない) CRCエラーなどが発生しているトラックの読み出しは行いません．

```
	}
}

void readfile(char *filename)		/* ファイル読み込み */
{
	FILE *fp;

	if((fp = fopen(filename, "rb")) == NULL) {
		fprintf(stderr, "ファイルがオープンできません.¥n");
		exit(1);
	}
	filelen = fread(buffer, 1, BUFFLEN, fp);
	fclose(fp);
}

void writefile(char *filename)		/* ファイル書き込み */
{
	FILE *fp;

	if((fp = fopen(filename, "wb")) == NULL) {
		fprintf(stderr, "ファイルがオープンできません.¥n");
		exit(1);
	}
	fwrite(buffer, 1, datalen, fp);
	fclose(fp);
}

void main(int ac, char *av[])			/* main関数 */
{
	unsigned char far *p;		/* メモリアクセス用farポインタ */
	char fname[80];			/* ファイル名 */
	int stat;			/* BIOSリターンコード */

	if(ac<3 || av[1][1]!=':' || av[1][2]!='¥0') {
		usage();
		exit(0);
	}

	track = atoi(av[2]);			/* トラック番号 */
	if(track<0 || track>163) {
		fprintf(stderr, "トラック番号が不正です.¥n");
		exit(0);
	}

	if(ac>3) {				/* /Wスイッチがついているか? */
		if((av[3][0]=='/' || av[3][0]=='-') && toupper(av[3][1])=='W'
		   && av[3][2]=='¥0')
			writeflag = 1;		/* 書き込みフラグ */
		else {
			fprintf(stderr, "オプションの指定が不正です.¥n");
			exit(0);
		}
	}

	p = (void far *)0x0000066cL;		/* ドライブ名→PDAテーブル */
	pda = p[toupper(av[1][0]) - 'A'];
	if ((pda & 0x70)!=0x70 && (pda & 0x70)!=0x10) {
		fprintf(stderr, "ドライブの指定が不正です.¥n");
		exit(0);
	}
	sprintf(fname, "%c_%.3d.dat", av[1][0], track); /* ファイル名作成 */

	drivecheck();		/* ドライブの状態をチェックする */
	allocate();		/* バッファにメモリブロックを割り当てる */

	if(writeflag==1)
		readfile(fname);	/* ファイルからバッファへ読み込む */

	dens = 1;
	if(idcheck()!=0) {				/* 2HD 倍密 */
		dens = 0;
		if(idcheck()!=0) {			/* 2HD 単密 */
			pda ^= 0x80;
			dens = 1;
			if(idcheck()!=0) {		/* 2DD 倍密 */
				dens = 0;
				if(idcheck()!=0) {	/* 2DD 単密 */
					fprintf(stderr, "読めません.¥n");
					exit(1);
				}
			}
		}
	}

	printf("¥nデバイス種別 : 2%cD ", (pda&0x80)==0 ? 'D' : 'H');
	printf("%s密¥n", dens==0 ? "単" : "倍");
	printf("トラック番号 : %d¥n", track);
	printf("セクタ長     : %d(%Xh)bytes¥n", 0x80<<id_n, 0x80<<id_n);
	printf("セクタ数     : %d¥n", max_r);
	datalen = (0x80<<id_n) * max_r;
	printf("データ長     : %d(%Xh)bytes¥n", datalen, datalen);
	printf("ファイル名   : %s¥n¥n", fname);

```

プログラムの説明

❖sense()

DISK BIOSのsense statusコマンドを使い，ドライブの状態(メディアの挿入とライトプロテクト)を調べます．

❖readid()

DISK BIOSのread IDコマンドを使い，指定されたトラック(の現在のヘッド位置)から1つIDを読み出します．

❖rw_data()

DISK BIOSのread data/write dataコマンドを使い，現在ヘッドのあるトラックでデータの読み書きを行います．int86x関数ではバッファのアドレス指定に使用するBPレジスタを操作できないので，インラインアセンブラを使ってDISK BIOSを呼び出しています．

❖idcheck()

指定したデバイスモード(2HD/2DD)と記録密度(単密／倍密)でトラックのIDが読み出せるかどうかをチェックし，読み出せる場合はIDのC，H，Nと最大セクタ番号を調べます．

❖drivecheck()

ドライブの状態を調べ，ディスクが

```
251:        if(datalen > BUFFLEN) {
252:                fprintf(stderr, "特殊なトラックです.Yn");
253:                exit(1);
254:        }
255:
256:        if(writeflag==0) {      /* トラックからファイルへ読み出し */
257:                stat = rw_data(READ_DATA);
258:                if(stat==0x10)          /* デリーテッドデータ */
259:                        stat = rw_data(READ_DELETED_DATA);
260:                if(stat!=0) {
261:                        fprintf(stderr, "リードエラー[%x]Yn", stat);
262:                        exit(1);
263:                }
264:                writefile(fname);
265:                printf("トラック内容をファイルに読み出しました.Yn");
266:        } else {                /* ファイルをトラックへ書き込み */
267:                if(datalen!=filelen) {
268:                        fprintf(stderr, "ファイルの長さが不正です.Yn");
269:                        exit(1);
270:                }
271:                stat = rw_data(WRITE_DATA);
272:                if(stat!=0) {
273:                        fprintf(stderr, "ライトエラー[%x]Yn", stat);
274:                        exit(1);
275:                }
276:                printf("ファイルをトラックへ書き込みました.Yn");
277:        }
278: }
```

挿入されていなかったり，書き込みの指定があるのにディスクがライトプロテクト状態のときはエラー終了します.

usage()

プログラムの使い方を表示します.

allocate()

データのリード／ライト用バッファにメモリを割り当てます．最初に割り当てたバッファが64KBのメモリ境界にかかっているとDMA転送の際にエラーが起こるので，再度メモリ割り当てを行います.

readfile()

指定したファイル名のファイルの内容をバッファへ読み出します.

writefile()

指定したファイル名のファイルを作成し，バッファ内容を書き込みます.

main()

191-194行

パラメータが少ないか，第1パラメータ(ドライブ名)が不正なら使い方を表示して終了します.

196-200行

第2パラメータをトラック番号に変換し，0より小さかったり163より大きければエラー終了します.

202行

「/W」スイッチ（/w，-W，-wでもよい）が指定されていれば書き込みフラグをセットします.

212-217行

MS-DOSのワークエリアを見て指定されたドライブ名からデバイス番号とユニット番号（PDA）を取得します(フロッピーディスク以外のドライブであればエラー終了).

218行

ドライブ名とトラック番号からファイル名を生成します.

226-240行

2HD倍密，2HD単密，2DD倍密，2DD単密の順にIDを読み出し，デバイスモードと記録密度を調べます．すべて読み出せなければフォーマットされていないトラックとしてエラー終了します.

242-249行

調べたデバイス種別やセクタ長，セクタ数などの情報を画面表示します.

251-254行

セクタの合計が10Kを越える場合(通常ではあり得ない)はエラー終了します.

256-265行

読み出しモード(/Wが指定されないとき)ではすべてのセクタをバッファへ読み出し，正常に読み出せたらファイルへ書き込んで終了します.

266-278行

書き込みモード(/Wが指定されたとき)ではバッファに読み込んだファイルの長さと調べたトラックのセクタの合計長を比べ，一致していればバッ

ファの内容をトラックへ書き込んで終了します．

ワンポイントテクニック

DISK BIOSを使う

MS-DOSのシステムコールには直接ディスクへアクセスするAbsolute Disk Read/Write (int 25h/26h) が用意されていますが，MS-DOSフォーマット(と物理フォーマットが一致する)のディスクしか読み書きすることができません．今回のプログラムでは，DISK BASICなどの物理フォーマットがMS-DOSフォーマットとは異なるフロッピーディスクを読み書きするため，DISK BIOSを利用してディスクへアクセスしています．BIOSはROM内ルーチンですから，PC-9801シリーズ（と互換機）に機種依存したものであることに注意してください．

PC-9801のDISK BIOSはFDC(フロッピーディスクコントローラ μPD765A）の機能をかなり自由に使えるようにできています．

DISK BIOSのコマンドを使いこなすにはフロッピーディスクのフォーマットについての知識が必要ですが，ここでは今回のプログラムで使ったコマンドについて説明することにします(SASIおよびSCSIのハードディスクに対してもDISK BIOSを利用して直接アクセスすることが可能ですが，コマンド体系は少々異なります)．プログラムとあわせて見てもらえれば，だいたいの動作が理解できるのではないかと思います．

DISK BIOSはint 1Bhで呼び出しますが，このときAHレジスタにコマンド識別コード，ALレジスタにデバイス識別コードを与えてやる必要があります．

AH, ALの各レジスタはビット毎に表1のような意味をもっています．

AHレジスタの下位4ビットで指定するコマンドには表2のようなものが

表2

0	seek
1	verify
2	read diagnostic
3	initialize
4	sense status
5	write data
6	read data
7	recalibrate
9	write deleted data
10	read ID
12	read deleted data
13	write ID

表1

●AHレジスタ（コマンド識別コード）

bit7	マルチトラック	1にするとリード／ライト系のコマンドでヘッド0からヘッド1へ連続して処理することができる．
bit6	記録密度	0で単密度，1で倍密度の記録方式を選択．
bit5	リトライ禁止	0にするとBIOS内部でリターンコード50H以上のエラーに対して7回までのリトライを行う．1でリトライ禁止．
bit4	シーク許可	0でシーク禁止，1でCLレジスタの示すシリンダにシークする．（CLレジスタを指定する必要のないコマンドは除く）
bit3-0	コマンド番号	

●ALレジスタ（デバイス識別コード）

bit7	アクセスモード	1で1MB（2HD），0で640KB（2DD）．
bit6-4	インターフェース	000:SASI HDD, 010:SCSI HDD, 001:1MB(2HD), 111:640KB(2DD).
bit2	ヘッド反転	0でDHレジスタの最下位ビット，1でそれを反転したビット側のヘッドにアクセスする（IDのHの最下位ビットと実際のヘッド位置が異なる場合に1にする）．
bit1-0	ユニット番号（SCSI HDDではbit3-0を使用）	

あります．

以下，今回のプログラムで利用した4，5，6，10の各コマンドについて説明します．

❖4 : sense status

ドライブの状態を調べます．AHレジスタに返されるリターンコードは

0xh……Ready
1xh……Write protected (書き込み禁止)
60h……Not ready（メディアが挿入されていない)

で，ドライブが両面モードであれば最下位ビットがセットされます．

❖5 : write data

ES:BP……データバッファ
BX……転送するデータの長さ
CL……IDのC（およびシークするシリンダ番号)
DH……IDのH（最下位ビットはヘッド指定)
DL……IDのR（セクタ番号)
CH……IDのN（セクタ長)

以上のパラメータにより，指定したセクタにデータバッファのデータを書き込みます．BXにセクタ長を越える値を指定すると，次のセクタ番号のセクタに続けて処理が行われます．

❖6 : read data

write dataと同様にパラメータを指定します．指定したセクタからBXバイト(セクタ長を越えるときは次のセクタ番号のセクタに続けて)分のデータがデータバッファに転送されます．

❖10 : read ID

CL……シークするシリンダ番号
DH……最下位ビットでヘッド指定

現在のヘッド位置から(シークが必要なときはシーク後)最初に見つかったIDのCHRNをそれぞれCL，DH，DL，CHレジスタに返します．

sense status以外のコマンドでは，AHレジスタにリターンコードが返され，20H以上のエラーが起きたときはキャリーフラグが立ちます．

表3　DISK BIOS リターンコード一覧

00H	正常終了	
10H	Control mark	read dataでDDAMが見つかったか，read deleted dataでDAMが見つかった
20H	DMA boundary	データバッファが64KB境界にまたがっている
30H	End of cylinder	EOT（データ長によって異なる）を越えて処理を続けようとした
40H	Equipment check	デバイスからfault信号を受け取ったか，recalibrateで一定時間内にtrack0信号が検出されなかった（このエラーが起こるときはハードが異常)
50H	Over run	non-DMAモードでデータ転送時にメインシステムが追い付かなかった（98ではDMA転送なので関係ない)
60H	Not ready	デバイスがready状態でなかった
70H	Not writable	ライト系のコマンドでwrite protect信号を検出した
90H	Time out	タイムアウト
A0H	Data error	ID CRCエラーを検出した
B0H	Data error	DATA CRCエラーを検出した
C0H	No data	目的のIDが見つからなかった
E0H	Missing ID address mark	インデックスホール信号を2回検出する間に1度もIDAMが見つからなかった
F0H	Missing DATA address mark	IDAMが見つかったあと，DAM（またはDDAM）が見つからなかった

第4回

イカすアレンジ編

酒井法雄

先月までで，構造化についての基本事項をひと通り紹介しました．しかし，実際にイカすプログラミングをするには，基本事項をどう応用するかにかかってきます．今月は，応用のためのノウハウを紹介しましょう．

すげぇぜMIDI

ある朝目覚めたら，私はどうしようもない不幸な気持ちに支配されていました．このままだと，どんどん不幸な気持ちになってしまいます．なんとか幸福を見つけなくては……．私は迷わず秋葉原に行きました．そして，幸せ探しウサギのように幸福になれそうなモノを探したのです．

いろいろな店をめぐっているうちに，ふと私はMIDIがやりたくなったのです．そう，コンピュータを使った音楽です．これなら，きたるべきマルチメディアのための基礎知識となり，仕事にも役立つだろうなどという勝手な言い訳も立ちます．

実は，それまでの私は，「MIDIなんてFM音源カードに毛が生えたようなオモチャ」とか，「パソコンで音を鳴らして何が面白いんだ」とか思っていました．しかし，このような偏見は，実際にMIDIに触れるとすべて崩れ去ってしまいました．

MIDIはオモチャではありませんでした．ある程度の値段のする音源は，実にいい音がします．それもそのはず，最近のMIDIの音源はPCMでサンプリングしたデータが入っているのです．その音を組み合せたり，数々のエフェクトをかけたりすることもできます．さらに，16トラックのマルチトラックレコーダを内蔵し，同時に32音もの音が出るというキーボードまであります．それが，パソコンより安い値段で買うことができるのです．

実際，CMやドラマの音楽などは，時間とコストのかからないMIDIで作られていることがほとんどです．その道の人が聞けば，「これはあの音源の音だ」とわかるくらいです．また，アーティストによっては，ほとんどMIDIだけでバックの音を作っている人も多いようです．

私は，ミュージックワークステーションなるキーボードと，一般的なパソコン用の音源，そしてパソコン用のシーケンサーソフトを買いました．せっかく高い機材を買ったのですから，やっぱり自分でも音楽を作ってみたいと思うようになりました．というわけで，キーボード(パソコンのではない)の練習をしはじめました．練習の手助けに，パソコンのシーケンサーソフトを使っています．入力したデータをリズムに使ったり，合奏したりと，結構役立ちます．もっとも，成果の方はあまりあがっていませんが……．

ポイントはアレンジだ

そんなある日，友人がパソコン通信などで流れているMIDIのデータを大量にくれました．私は喜んで，次々と演奏させてみました．しかし，そのほとんどはなんとも味気ないモノ（失礼！）でした．たしかに，その曲とわかるモノではありますが，実に大味な音楽です．いかにも「パソコンが演奏している」という感じです．ドラマやCMで使われているMIDIで作られた音楽とは月とスッポンです．まぁ，相手はプロですから，違ってあたり前ではありますが…….

どうしてなのだろうかと，私は考えてみました．まず，音の選びかたがよくありません．また，強弱にも乏しく，ビブラートを効かせたりというようなコワザもありません．そう，アレンジが悪いのです．

もっとも，このようなデータを手作業で入力している人の多くは，楽譜を見ながらデータを打ち込むだけで，気力を使い果たしてしまうのでしょう．自分で曲を作る人も，なかなかアレンジまでしっかり凝って作る余裕はないのかもしれません．

あれ？　これってどこかで聞いたような話じゃありませんか？　そう，これはまさにパソコンのプログラミングと同じなのです．雑誌や本のリストを打ち込んで，「動く動く」といってとりあえず満足している人．自分でプログラミングして動くことは動くけれど，どうもイマイチだと思っている人．そう，いま一歩足りないのは，アレンジなのです．

プログラミングでのアレンジには，大きく分けて2つの要素があります．1つは，ちゃんとカッコよく動作するということです．もう1つは，ソースリストが見やすくてイカすことです．

この2つは別々のモノでありながら，密接な関係にあります．ソースリストがわかりやすく構造化してあれば，バグも出にくく，出ても取りやすいのです．そして，あとからの仕様変更にも柔軟に対応できます．つまり，まずはイカすソースプログラムを書くことを最初の目標とすればいいわけです．まさに，この連載のテーマです．

先月までに，構造化制御構文とプロシージャについてひと通りの解説をしました．そこで，今月は，これらをどうアレンジして，イカすプログラムに仕上げるかについてお話ししましょう．

プロシージャは構造化と仲良し

「構造化制御構文は，一応使えます．プロシージャも作れますが，どうもうまく使いこなせません．まとまらないんです．」このような話をよく聞きます．これは，典型的なアレンジができていない例です．ひと通りの知識があるのに実践できないのは，そのご利益を本当には理解できていないからです．

では，ご利益とは何でしょうか？　何度もいっていることですが，それは「見通しのよいわかりやすいプログラミングができる」ことです．このご利益は，構造化制御構文とプロシージャのいずれにもあることです．さらに，この2つを組み合わせると，さらなるご利益が出てきます．問題は，その組み合わせ方です．

ここで，カツ丼を作ることを考えてみましょう．カツ丼は，ご存知のように代表的な丼モノです．カツ丼の構成要素としては，タレ，タマゴ，カツ，ごはん，どんぶり，グリーンピース，きざみ海苔などがあります．店によっては，グリーンピースがなかったり，ほかのモノがあったりするかもしれませんが，ここではこのようなカツ丼を考えます．

場末の安い定食屋でカツ丼を作る風景を見ていると，その手際よさに驚いてしまいます．まず，カツを適当な大きさに切り，底が平らなフライパンのようなカタチをしたモノに入れます．タマゴをかきまぜて，タレと一緒に上からかけます．その上にグリーンピースを載っけて，フタをして火にかけます．その間に，丼にごはんを盛り付けます．具ができたら，ごはんの上にかけて，きざみ海苔をふりかけて，できあがります．この間，ホンの数分です．

では，私たちがカツ丼を作ろうとすると，どうするでしょうか？　カツはどうしますか？　惣菜屋さんから買ってきますか？　それとも，自分で揚げますか？　タレはどうしますか？　できあいのビン入りのタレを使いますか？　それとも，カツオ節を削り出すところからやりますか？　ノリはきざみ海苔を買ってきますか？　それとも，グルメを気どって三陸の高級海苔を自分で焼いて刻みますか？　いろいろなやり方があるでしょうが，どれにしてもできあがる早さは場末の定食屋のオヤジにはかなわないでしょう．それは，なぜでしょうか？

場末の定食屋のオヤジさんは，ちゃんと材料を仕込んであります．カツはあらかじめ揚げてありますし，タレも作ってあります．海苔もきざみ海苔が用意されています．ごはんだって大量に炊いてあります．そう，この仕込みこそ早くできる武器なのです．

ワンラインNEWS
ロータスは表計算ソフト「1-2-3」のマルチプラット戦略が完了したとし，年内にもMS-Windows 3.0に対応したワープロ「Amipro」を日本語化し，出荷する．

仕込みは，そのままプロシージャと考えることができます．もし，QuickBASICに「カツ」とか「タレ」とか「ごはん」というプロシージャがあらかじめ用意されていれば，私たちはカンタンにカツ丼というプログラムを作ることができるでしょう．しかし，現実には用意されていません．ならば，自分で作りましょう．それが，SUBやFUNCTIONといったプロシージャの重要な用途なのです．

バーチャルプログラム「カツ丼」

ここに，「カツ丼.EXE」というプログラムがあります．ちゃんと湯気が出ているカツ丼のバーチャルプログラムです．ラッキーなことに，「カツ丼.BAS」というソースプログラムもあります．では，早速中身を見てみましょう．

メインプロシージャは，次のようになっています．

```
PRINT カツ丼 ()
```

なんと単純でわかりやすいメインプロシージャでしょう．しかし，これだけでは中の処理がわかりません．次に，カツ丼プロシージャを見てみましょう．

```
FUNTION カツ丼
    TMP = 盛り付ける(丼, ごはん)
    TMP = 盛り付ける(TMP, カツ丼の具)
    カツ丼 = かける(TMP, きざみ海苔)
END FUNCTION
```

これまた，単純ですね．盛り付けるプロシージャは，1つめの引数に2つめの引数を盛り付けるものです．カツ丼とはなんぞやといったら，まさにこの通りです．では，カツ丼の具は……

```
FUNTION カツ丼の具
    TMP = スライス (カツ) +
        タレ +
        かきまぜ (タマゴ) +
        グリーンピース
    DO
        煮る(TMP)
    LOOP UNTIL 煮えた
    カツ丼の具 = TMP
END FUNCTION
```

なるほど．これでカツ丼の具ができるわけです．スライスプロシージャや，かきまぜプロシージャ，煮るプロシージャなどは，名前の通りのことをやるのでしょうから，あまり見る必要はなさそうです．では，カツはどうやって作っているのでしょうか？

```
FUNCTION カツ
    TMP = 衣 (スライス (豚肉))
    DO
        揚げる (TMP)
    LOOP UNTIL 中まで火が通る
    カツ = TMP
END FUNCTION
```

てんぷらでもないのに，TMPとはこれいかに……．つまらない冗談はさておき，これもわかりやすいですね．

このように，カツ丼を作るというプログラムでは，実にいろいろな作業が行われているのですが，それぞれの作業をブロック化し，下ごしらえしておけば，見通しのよいカツ丼の作り方がわかるわけです．

カツ丼プロシージャで行われている実際の作業は，図1のような構造になります．

実際には，ここに書いていないようなプロシージャもあるでしょう．また，ここに書いてあるような面倒なことをしない方法もあるでしょう．たとえば，「カツ丼の具」というレトルトパックを買ってくれば，このような面倒な作業のほとんどが省けます．しかし，よく考えてください．それは「カツ丼の具」というプロシージャなのです．このよ

▼図1　カツ丼プロシージャ

```
カツ丼
        丼
                どんぶりを買ってくる
        カツ丼の具
                カツ
                        衣(スライス(豚肉))
                        揚げる
                タレ
                        煮る(醤油 + みりん + 鰹節)
                タマゴ
                        鶏を育てる
                        タマゴを生ませる
                グリンピース
                        グリーンピース缶詰を買ってくる
                煮る()
        ごはん
                米を買ってくる
                とぐ
                炊飯
                保温
        きざみ海苔
                海苔を買ってくる
                焼く
                刻む
        盛り付ける
```

うに，みんなが欲しがりそうなプロシージャは，市販されていることもあります．そのようなモノを，ライブラリと呼んでいます．QuickBASICには，データベースにアクセスするためのISAMライブラリや，画面を簡単に作ることのできるライブラリなどが市販されています．このようなライブラリは，効率よくプログラミングする手助けとなるでしょう．

イカすプログラムのための2つの方針

このように，複雑な作業をブロック化して考え，その構造を明確にすることが大切です．つまり，イカすプログラムを作るには，

①手順を構造化制御構文で明確にする
②ひとまとまりの作業をプロシージャにする

という2つの作業がもっとも重要なのです．

実際には，②は後で考えます．最初に，「こんなプロシージャがあったらいいな」と思ったら，そのプロシージャの仕様の概略（引数，戻り値など）を考え，「あるものと仮定し」ます．そして，おおまかな処理の流れを書いてしまい，「さっさと全体像をはっきり」させます．カツ丼の例でいえば，「FUNCTION カツ丼」をさっさと作るということです．

ここで，「あったらいいな」というプロシージャは，単機能であまり特殊でない方が望ましいです．たとえば，「カツ丼の具」と「ごはん」を1つのプロシージャにした場合を考えましょう．あとから，「天丼」を作りたいと思ったとき，このプログラムはツブシがききません．最初から「カツ丼の具」と「ごはん」に分けてあれば，「天丼の具」プロシージャを作って，「カツ丼の具」と入れ換えるだけです．

また，あとから「カツライス」プログラムを作ろうしたとき，「カツ」プロシージャはそのまま使うことができるでしょう．「煮る」，「揚げる」，「盛り付ける」などのプロシージャは，いろいろな場面で使われます．もちろん，バーチャル中華料理プログラムを作るときにも役立つでしょう．「カツを煮る」だけのプロシージャでは，ツブシがききません．いろいろな場面で使えるプロシージャは，なるべく汎用に作りましょう．それは資産となって，別のプログラムを作るときにも自分だけのライブラリとして役立てることができます．

このように，最初に全体の流れを作り，その下のプロシージャを順に作っていくプログラミングのやり方を，「トップダウン手法」といいます．このトップダウン手法を使うと，意外なほどラクにプログラミングができます．というのも，構造化制御構文やプロシージャという概念は，そもそもがトップダウン手法を支援するために考えられたモノだからです．

ブロックの作り方

トップダウン手法を使って，上から順にプログラムの構造を作っていくとき，一番困るのは「ブロック」の切り分けでしょう．つまり，「どこからどこまでを，ひとつのプロシージャにするか」ということです．逆に，これができるようになれば，トップダウン手法も構造化制御構文も，プロシージャも自由自在に扱うことができます．

しかし，そのプログラムの内容によっては，この切り分けは非常に難しいものになることもあります．実行速度を優先するか，メンテナンス性を優先するかなどによっても，切り分けは変わってきます．しかし，「わかりやすいプログラミング」を最優先にすべきです．いくら速くても，あとでメンテナンスできないプログラムでは困ります．

では，どのような切り分け方が，メンテナンス性をよくするのでしょうか．ここでは，カンタンに3つにまとめてみました．

1．汎用プロシージャを作る

先ほども述べましたが，よく使う処理をプロシージャにしておけば，自分のライブラリとして後から使いまわすことができます．たとえば，座標，文字列，色を引数とした表示ルーチンなどを作っておけば，毎回「LOCATE」，「COLOR」，「PRINT」などを何度も連続して書く必要がなくなります．

2．作業の内容によってブロック化する

「ここは，○○の処理をするところ」という考え方です．これは，そのプログラムによって特有な処理で構いません．たとえば，「ファイルからの読み出し」を1つのプロシージャにしてしまえば，ファイルから正常に読み出せていないときは，そのプロシージャに問題ありと，問題点が明らかにしやすくなります．N88BASICなどの「GOSUB」文の行き先だったところと同じように考えればいいでしょう．

3．似た処理をまとめる

ワンラインNEWS
日本シリコングラフィックスはグラフィックス性能を向上させたエンジニアリンググラフィックWS「IRIS Indigo」シリーズ3機種を発売した．

作業の内容によっては,「かなり似ている処理でもちょっとだけ違う」ということがあります. たとえば,「ファイルの中に"Yoriko"という文字列があったら, そのレコード番号を返す」と「ファイルの中に"Yoriko"という文字列がないことを確認する」処理は, 一見すると違うようですが, 実はほとんど同じです. このような処理は,「あったらそのレコード番号を, なかったら0を返す」というプロシージャにまとめることができます. このように, 似たような処理を「アタマを使ってまとめる」と, プログラムも短くなりますし, 後から修正したいときも, 1か所ですみます.

実際のプログラミングでは, いろいろな場合が出てきます. 最初からうまく切り分けできないかもしれませんが, それも勉強です. 徐々にスキルを上げていきましょう.

見やすいプログラミングのために

構造化とプロシージャの基礎的な解説はこれで終りです. しかし, ホントにイカすプログラミングには, これだけでは足りません. いかに, 読みやすいプログラムであるかという問題があります. 読みやすいということは, デバッグもしやすくなります. また, あとになってプログラムを改造しようと思い立ったとき,「ここは何をやっているんだ?」などと悩む時間も少なくてすみます.

そのポイントを次にまとめます.

1. 定数を使おう

QuickBASICでは, 定数を使うことができます. 変数と違って定数は, 1度決めたら内容を変更することができません. ですから, 画面の座標や最大レコード数など, 1度プログラムを動かしたら変更が必要ないものは, 定数にしましょう. 変数にすると, まちがって変な数を代入して動作がヘンになるということもあります.

また, 定数はプログラムの最初にまとめて宣言しておきましょう. プログラムの中で, 定数に支配されるような数は, 定数を元に計算するようにしましょう. こうすれば, 後から変更したいときも1か所の変更だけですみます.

定数と変数を区別するため, 定数は大文字にしましょう. また, 内容がわかりやすいような定数名にしましょう.

2. 変数は宣言しよう

QuickBASICでは, 変数を宣言することができます. 急いでプログラミングするときには, 突然その場で変数を使えるのはBASICのメリットともいえますが, バグの原因にもなります. 使う変数は, 必ず宣言するクセをつけましょう.

定数と変数を区別するために, 変数は小文字にしましょう. もちろん, 内容がわかりやすいような変数名にしましょう.

3. プロシージャの命名法

プロシージャを漫然と使っていると, 変数や定数と区別できなくなります. プロシージャは, 大文字と小文字を取り混ぜて, 処理内容のわかりやすい名前にしましょう.

4. インデントをつけよう

構造をわかりやすくする構造化制御構文も, だらだらと書いていてはブロックがはっきりしなくてわかりにくいものになってしまいます. IFやLOOPなどのブロックは, その範囲が明確になるようにタブを使ってインデントをつけましょう.

5. プロシージャは短く

プロシージャは, 長くなりがちです. 処理をわかりやすく区切るためのプロシージャなのですから, プロシージャどうしの構造をうまく考えて, なるべく短いプロシージャになるように心がけましょう.

基礎編終了

さて, これでQuickBASICでイカすプログラミングをするための基礎は, ひと通り解説しました. 今月は概念的な話ばかりで, 少々退屈だったかもしれません. 来月からは, いよいよ実戦編です. いままでの基礎知識を元にして, まとまったプログラムを作ってみましょう.

ところで, 友人から私のもとに妙な本が届きました. なんと, MIDIインターフェースの資料です. これでドライバとシーケンサーのプログラムを作れということなのです. 私は思いました.「これは, 新たな不幸のはじまりかもしれない」と.

ワンラインNEWS
米ネクスト・コンピュータはOSをハードウェア販売と切り離し, 市販パソコン向けに供給する. これまで同社のOSは自社製のWSでしか動かなかった.

Illustration : Masami Nozue

アルゴリズム篇 第4回
サーチ・その2

実践C塾

三代目塾頭
㈱ジャステックラボ
加野島英渡／山本博昭

先月は，ランダムなデータ列を対象とした順次検索（シリアルサーチ）について解説したが，今月はランダムでないデータの場合のより効率のよい検索方法について解説を行う．今回とりあげるアルゴリズムは，バイナリサーチである．

先月までのあらすじ

前回は，テーマツールの「単語リスト作成」の処理部分に着目して，「単語が初出か既出か？」の判断をいかに効率よく行うかを考察してみた．テーマツールが扱っている単語リストは，ファイル内の単語を出現順に登録していくため，データ列の並びがランダムになっている．このような並びのデータ列から，目的のキーワードを検索するためには，順次検索(シリアルサーチ)手法でデータ列の先頭から１つずつ検索を行わなくてはならない．前回はこの順次検索について，発見したデータをデータ列の先頭方向へ移動して次回からの検索をスピードアップする手法や，データ列末尾に「番人」をおいて，終了判定を簡略化するといった手法を紹介し，それらの手法を使ってテーマツールの単語リスト構築部分を改造してみた．

今月の重点項目

先月の主眼は，「ランダムなデータ列を対象とするならば，順次検索（シリアルサーチ)を行わなくてはならない」ということであった．では，ランダムでないデータの場合なら，もっと効率のよい検索方法があるのだろうか？今月はこの疑問から考えていってみよう．

ここにさまざまな年齢の人間が100人集まっていて，その中から39才の人を探し出しなさいという問題があるとする．人の列が年齢順不同で並んでいたら，39才という人を発見するためには，先頭の人から順に歳を聞いていくしか方法はなさそうだ．これが順次検索である．ところが，列が前もって年齢の若い順に並んでいてくれれば，先頭のほうの赤ん坊や幼稚園児はおそらく39才ではないので，列のまんなかあたりの人から歳を聞きはじめることができる．だいたいの見当をつけて，これとおぼしき人に歳を聞いてみる．その人がじつは見た目より若くて「28才です」などというかもしれない．その場合は，さらに列の後ろのほうに進んでいく．後ろに行きすぎて「42才」などという声が聞こえたら今度は前のほうに戻る．このときに，さきほどの28才の人のいた場所より前に戻ることはない．年齢順に並んでいるのだから，28才の人の前に，39才の人がいることはありえないからである．こうして何度か前後に移動しながらその範囲を狭めていけば，39才の人間はほどなく見つかるという算段である．

データ列が，ある法則に従って整列済みであるということを前提にすると，いま書いたような検索方法が可能である．この手法のポイントは，データ列のある地点に存在するデータを検索対象となるデータと比較し，その結果によってその地点より前に移動するか後ろに移動するかを決定することである．ここで考えなければならないことがいくつかある．

最初に比較をはじめるのは，どこからか．

データ列の分布状態や検索対象データの内容によって，検索効率は変化してくるが，汎用的なアルゴリズムとして考えると，データ列の「まんなか」の位置から検索を開始するのがもっとも簡便な処理となるだろう．

次の比較地点をどこまで移動するか．

39才の人を探す例では，人相風体を見ながら進んだり戻ったりできるかもしれないが，プログラムのアルゴリズムではそういうわけにはいかない．最初に比較をはじめた地点がデータ列の「まんなか」であれば，次の比較地点はそのまた半分，つまり前または後ろの1/4地点としよう．

基本的な考え方

整列済みのデータを検索するときの考え方を，図１を見ながら整理してみよう．

①データ群は，すでに整列済みであること（以降の説明は，小さいものからしだいに大きいものへと整列していることを前提とする)．
②データ群の「まんなか」の位置のデータと検索対象データを比較する．
③比較の結果，検索対象データのほうが小さければ，いまの位置以降には検索対象データと等しいものはあり得ないので，「まんなか」から左側のデータ群を対象にして，その「まんなか」の位置のデータと検索対象データの比較を行う．
④比較の結果，検索対象データのほ

うが大きければ，いまの位置以前には検索対象データと等しいものはあり得ないので，「まんなか」から右側のデータ群を対象にして，その「まんなか」の位置のデータと検索対象データの比較を行う．

上記のような処理を繰りかえすと，1回の比較でデータ群の中の検索範囲が半分ずつになってくる．そして，範囲が1になっても発見できない場合は，データ群の中に目的のデータが含まれていないということになる．データ群を2分していきながら検索範囲をせばめていくこの手法は，「バイナリサーチ」と呼ばれる．

バイナリサーチの具体例

今度は，具体的な数値を使ってバイナリサーチをシミュレートしてみよう．図2のように1から15までの数値が昇順に並んだデータ群から，「11」を検索対象データとして検索する処理を机上で行ってみることにする．

図2の例は，バイナリサーチを進めていって，データ群が1になったときに発見された場合，つまりバイナリサーチの最悪のケースであるが，それでも3回の比較で終了する．この検索をシリアルサーチで行うと，当然のことながら11回の比較が必要になる(通常，ソート済みのデータにたいしてシリアルサーチを行うという暴挙はあり得ないはずだが……)．

もちろん，検索対象データとして1や2を検索する場合には，シリアルサーチのほうが速いことになるが，ほとんどのケースで，バイナリサーチはシリアルサーチより高速になることがおわかりいただけたであろう．

図1　バイナリサーチの考え方

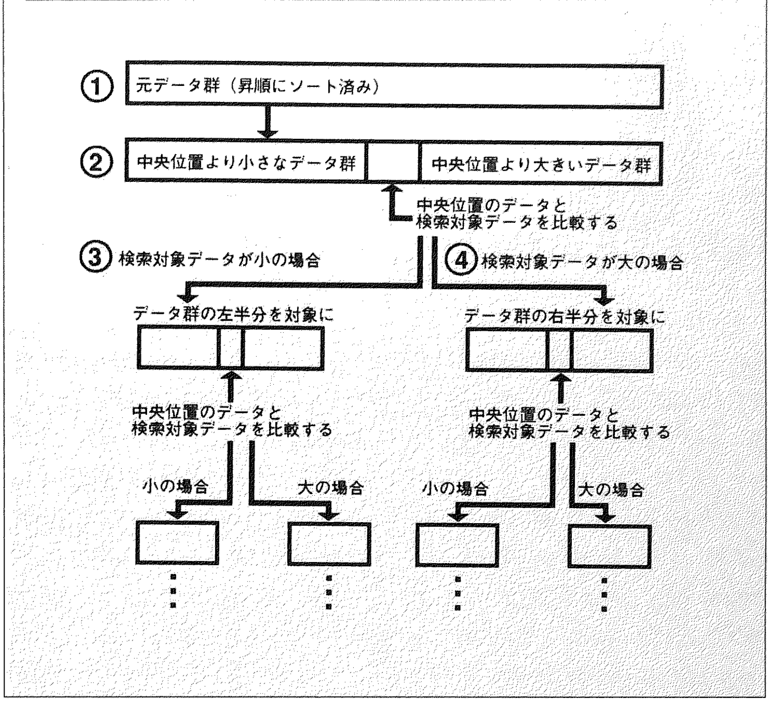
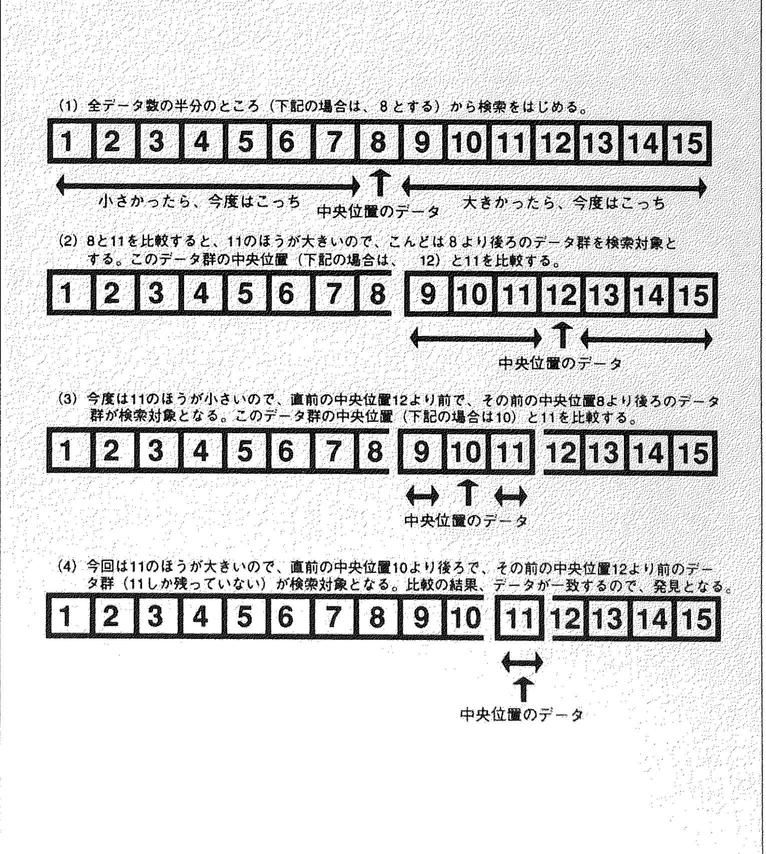


図2　バイナリサーチの具体例



ワンラインNEWS
日本経営協会のまとめた「家庭におけるパーソナルOA機器の普及状況・利用動向に関する実態調査」によると，ファミコンなどテレビゲーム機の普及率は50%弱.

バイナリサーチのコーディング例

では，実際にバイナリサーチのロジックをCを使って書いてみよう．リスト1と2が，バイナリサーチのテスト用に書いたサンプルプログラム「BSEARCH.C」とそのMAKEファイルである．このサンプルプログラムは，1から7まで昇順にならんだデータ群の中から，0から8までの検索対象データをバイナリサーチで次々に検索する処理を行う．例によってBorland C++でのコーディングであるが，MS-DOSのC処理系であればたいていはコンパイルすることができることと思う．起動はオプションなしに「A> BSEARCH <ret>」とする．現状のままコンパイルすると，テスト用ロジックによって途中経過を表示しながら処理するようになっている．

バイナリサーチ部分のコーディングのポイントは，再帰呼び出しである．中央位置と検索対象データの比較で結果が等しくない場合には，その中央位置の右側か左側のデータ群のポインタを新しいデータ群として，自分自身に引き渡して再検索する．

具体的なコーディングの解説は，核の部分となるバイナリサーチ関数「bsearch」の動作説明として図3にまとめた．

このサンプルプログラムを実行した画面が図4である．条件コンパイルによって組み込まれたテスト用のロジックで，いまどこを検索しているかの経過が表示されている．これを図5を見ながら検証してみることにしよう．実際に比較を行うデータは，ベースとなるポインタbptrから中央位置dmidだけ隔たった位置になる．つまり，そのアドレスは，(bptr + dsize * dmid) である．

リスト1　BSEARCH.MAK

```
.AUTODEPEND

#                    *Translator Definitions*
CC = bcc +BSEARCH.CFG
TASM = TASM
TLINK = tlink

#                    *Implicit Rules*
.c.obj:
   $(CC) -c {$< }

.cpp.obj:
   $(CC) -c {$< }

#                    *List Macros*

EXE_dependencies =  ¥
   BSEARCH.obj

#                    *Explicit Rules*
BSEARCH.exe: BSEARCH.cfg $(EXE_dependencies)
   $(TLINK) /x/n/c/P-/LA:¥BC¥LIB @&&|
c0s.obj+
BSEARCH.obj
BSEARCH
                     # no map file
emu.lib+
maths.lib+
cs.lib
|

#                    *Individual File Dependencies*
BSEARCH.obj: BSEARCH.c

#                    *Compiler Configuration File*
BSEARCH.cfg: BSEARCH.mak
   copy &&|
-ff-
-vi
-IA:¥BC¥INCLUDE
-LA:¥BC¥LIB
-P-.C
| BSEARCH.cfg
```

リスト2　BSEARCH.C

```
/* ==================================================================

     検索アルゴリズムチェック用サンプル "BSEARCH.c"
                  Copyright 1992 by JUSTECH-LABO. (H.kanoshima)

=================================================================== */

#include    <stdio.h>

#define     TEST    1

/* ======================================================================


======================================================================= */
void    *bsearch(void   *key,
                 void   *bptr,
                 int    dnum,
                 int    dsize,
                 int    (*compar)(const void *, const void *))
{
int     dmax, dmid;
int     res;
char    *midptr;

     if(dnum == 0)   return  NULL;
     dmax = dnum - 1;
     dmid = dmax / 2;
#if TEST
     printf("bptr = %04X dsize = %02d dmid = %02d¥n",
                                                 bptr, dsize, dmid);
#endif
     midptr = (char *)bptr + dsize * dmid;
     res = (*compar)(key, midptr);

     if      (res == 0)
          return  midptr;
     else if (res <  0)
          return  bsearch(key, bptr,          dmid,        dsize, compar);
     else
          return  bsearch(key, midptr + dsize, dmax - dmid, dsize, compar);
}

/* ======================================================================


======================================================================= */
int     basedata[] = {1,2,3,4,5,6,7};

int     icomp(const void *x, const void *y)
{
const   int     *xx = x;
const   int     *yy = y;

     return *xx - *yy;
}

/* ======================================================================


======================================================================= */
void    main()
{
int     i;

     for(i = 0; i <= 8; i++)
     {
          printf("%02d-%04X¥n¥n",i, bsearch(&i, basedata, 7, sizeof(int), icomp));
     }
}
```

ワンラインNEWS
トマスは横河・ヒューレット・パッカードのUNIXサーバ「HP9000シリーズ700」用の高速大容量テープバックアップシステム「Herstal8500A」を発売した．

図3 サンプルプログラム「BSEARCH」のバイナリサーチロジック

```
: void    *bsearch(void    *key,
引数1：検索対象のキーワードとなるデータへのポインタ

:                 void    *bptr,
引数2：データ群へのポインタ

:                 int     dnum,
引数3：データ群の要素数

:                 int     dsize,
引数4：データ群の1要素のバイト数

:                 int     (*compar)(const void *, const void *))
引数5：比較関数へのポインタ

: {
: int         dmax, dmid;
: int         res;
: char    *midptr;
:     if(dnum == 0)    return    NULL;
データ群が空ならば，何もしないで終了する．

:     dmax = dnum - 1;
検索するデータ最大位置の初期値

:     dmid = dmax / 2;
検索するポイントの初期値

: #if    TEST
:     printf("bptr = %04X dsize = %02d dmid = %02d¥n",
:                                          bptr, dsize, dmid);
: #endif
テスト用のロジック，サーチの途中経過（現在検索しているポインタ）を表示する．

:     midptr = (char *)bptr + dsize * dmid;
比較位置を算出する．

:     res = (*compar)(key, midptr);
比較を実行する（結果は，
                    負＝キーの方が小さい
                    0＝等しい
                    正＝キーの方が大きい）

:     if       (res == 0)
:         return    midptr;
比較の結果が等しければ，データが見つかったことになる．

:     else if    (res <  0)
:         return    bsearch(key, bptr,           dmid,         dsize, compar);
キーの方が小さければ，ベースとなるポインタから今チェックしたデータまでを対象に再び検索を行う．

:     else
:         return    bsearch(key, midptr + dsize, dmax - dmid, dsize, compar);
キーの方が大きければ，今チェックしたデータの1つ後ろからデータ検索位置の最大値までを対象に再び検索を行う．

: }
```

図4 サンプルプログラム「BSEARCH」の実行画面

```
A>BSEARCH
bptr = 00A6 dsize = 02 dmid = 03
bptr = 00A6 dsize = 02 dmid = 01
bptr = 00A6 dsize = 02 dmid = 00
00-0000

bptr = 00A6 dsize = 02 dmid = 03
bptr = 00A6 dsize = 02 dmid = 01
bptr = 00A6 dsize = 02 dmid = 00
01-00A6

bptr = 00A6 dsize = 02 dmid = 03
bptr = 00A6 dsize = 02 dmid = 01
02-00A8

bptr = 00A6 dsize = 02 dmid = 03
bptr = 00A6 dsize = 02 dmid = 01
bptr = 00AA dsize = 02 dmid = 00
03-00AA

bptr = 00A6 dsize = 02 dmid = 03
04-00AC

bptr = 00A6 dsize = 02 dmid = 03
bptr = 00AE dsize = 02 dmid = 01
bptr = 00AE dsize = 02 dmid = 00
05-00AE

bptr = 00A6 dsize = 02 dmid = 03
bptr = 00AE dsize = 02 dmid = 01
06-00B0

bptr = 00A6 dsize = 02 dmid = 03
bptr = 00AE dsize = 02 dmid = 01
bptr = 00B2 dsize = 02 dmid = 00
07-00B2

bptr = 00A6 dsize = 02 dmid = 03
bptr = 00AE dsize = 02 dmid = 01
bptr = 00B2 dsize = 02 dmid = 00
08-0000
```

図5 比較の結果，検索対象データが小さかった場合に，次の検索のために自分自身に引き渡す情報

```
bptr = 00A6 dsize = 02 dmid = 03    ※4と比較する  ->  0の方が小さい
bptr = 00A6 dsize = 02 dmid = 01    ※2と比較する  ->  0の方が小さい
bptr = 00A6 dsize = 02 dmid = 00    ※1と比較する  ->  0の方が小さい
00-0000                             ※0はデータ群に含まれていない

bptr = 00A6 dsize = 02 dmid = 03    ※4と比較する  ->  1の方が小さい
bptr = 00A6 dsize = 02 dmid = 01    ※2と比較する  ->  1の方が小さい
bptr = 00A6 dsize = 02 dmid = 00    ※1と比較する  ->  等しい
01-00A6

bptr = 00A6 dsize = 02 dmid = 03    ※4と比較する  ->  2の方が小さい
bptr = 00A6 dsize = 02 dmid = 01    ※2と比較する  ->  等しい
02-00A8

bptr = 00A6 dsize = 02 dmid = 03    ※4と比較する  ->  3の方が小さい
bptr = 00A6 dsize = 02 dmid = 01    ※2と比較する  ->  3の方が大きい
bptr = 00AA dsize = 02 dmid = 00    ※3と比較する  ->  等しい
03-00AA

bptr = 00A6 dsize = 02 dmid = 03    ※4と比較する  ->  等しい
04-00AC

bptr = 00A6 dsize = 02 dmid = 03    ※4と比較する  ->  5の方が大きい
bptr = 00AE dsize = 02 dmid = 01    ※6と比較する  ->  5の方が小さい
bptr = 00AE dsize = 02 dmid = 00    ※5と比較する  ->  等しい
05-00AE

bptr = 00A6 dsize = 02 dmid = 03    ※4と比較する  ->  6の方が大きい
bptr = 00AE dsize = 02 dmid = 01    ※6と比較する  ->  等しい
06-00B0

bptr = 00A6 dsize = 02 dmid = 03    ※4と比較する  ->  7の方が大きい
bptr = 00AE dsize = 02 dmid = 01    ※6と比較する  ->  7の方が大きい
bptr = 00B2 dsize = 02 dmid = 00    ※7と比較する  ->  等しい
07-00B2

bptr = 00A6 dsize = 02 dmid = 03    ※4と比較する  ->  8の方が大きい
bptr = 00AE dsize = 02 dmid = 01    ※6と比較する  ->  8の方が大きい
bptr = 00B2 dsize = 02 dmid = 00    ※7と比較する  ->  8の方が大きい
08-0000                             ※8はデータ群に含まれていない
```

図6 比較の結果，検索対象データが大きかった場合に，次の検索のために自分自身に引き渡す情報

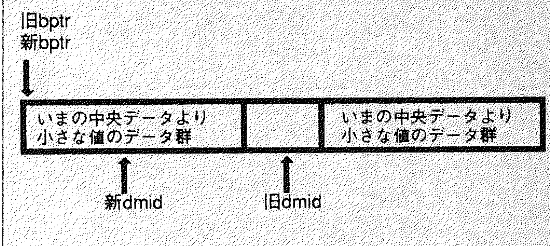
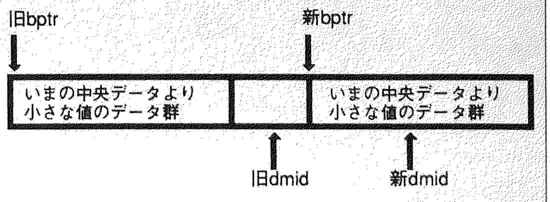


図7 サンプルプログラム「BSEARCH」の実行検証



ワンラインNEWS
日本ユニシスはデスクトップ型WS「S8000」シリーズの下位機種「S8100」を発表．モトローラのRISC型MPU「MC88100」を採用，1CPUで29MIPSを実現している．

比較の結果，検索対象データのほうが小さかったときは，dmid以前のデータ群のポインタ(現在のbptrと等しい)が検索対象となり，新しいdmidは現在のdmidの半分の値となる(図6)．

比較の結果，検索対象データのほうが大きかったときは，dmid以降のデータ群のポインタ(現在のdmidの1つ右のデータポインタ)が検索対象となり，新しいdmidは現在のdmidの半分の値となる(図7)．

上記のように，比較の結果によって，左半分または右半分のデータ群のポインタを自分に引き渡して検索を続行するため，dmidの値は結果にかかわらずつねに前回の2分の1となる．

バイナリサーチをCでコーディングした例を見てきたわけだが，核となる検索部分は，「検索を続行する場合は，新しい検索データ群とその範囲を自分自身に引き渡して，再帰的に処理をする」という手法をとった．今度はこの部分を再帰呼び出しを使わずにやってみよう．再帰呼び出しの場合は，新し

リスト3 BSEARCH2.C

```
/* ===================================================================

     検索アルゴリズムチェック用サンプル "BSEARCH2.c" 非再帰版
                  Copyright 1992 by JUSTECH-LABO. (H.kanoshima)

==================================================================== */

#include    <stdio.h>

#define     TEST    1

/* ========================================================================


========================================================================= */
void    *bsearch(void   *key,
                 void   *bptr,
                 int    dnum,
                 int    dsize,
                 int    (*compar)(const void *, const void *))
{
int     dmin, dmax, dmid;
int     res;
char    *midptr;

    dmax = dnum - 1;
    dmin = 0;

    while(dmin <= dmax)
    {
        dmid = (dmax + dmin) / 2;
#if TEST
    printf("bptr = %04X dsize = %02d dmid = %02d¥n",
                                              bptr, dsize, dmid);
#endif
        midptr = (char *)bptr + dsize * dmid;
        res = (*compar)(key, midptr);

        if      (res == 0)
            return  midptr;
        else if (res <  0)
            dmax = dmid - 1;
        else
            dmin = dmid + 1;
    }
    return  NULL;
}
/* ========================================================================


========================================================================= */

int     basedata[] = {1,2,3,4,5,6,7};

int     icomp(const void *x, const void *y)
{
const   int     *xx = x;
const   int     *yy = y;

    return *xx - *yy;
}

/* ========================================================================


========================================================================= */
void    main()
{
int     i;

    for(i = 0; i <= 8; i++)
    {
        printf("%02d-%04X¥n¥n",i, bsearch(&i, basedata, 7, sizeof(int), icomp));
    }
}
```

図8 再帰呼び出しを使わないでバイナリサーチを行うサンプル

```
:     dmax = dnum - 1;
検索するデータ最大位置の初期値

:     dmin = 0;
検索するデータ最小位置の初期値

:     while(dmin <= dmax)
:     {
データ最少位置と最大位置が重なる地点（検索すべきデータがなくなる）まで検索を行う．

:         dmid = (dmax + dmin) / 2;
比較を行う地点は，最小位置と最大位置の中間点である．

: #if    TEST
:     printf("bptr = %04X dsize = %02d dmid = %02d¥n",
:                                              bptr, dsize, dmid);
: #endif
テスト用のロジック．サーチの途中経過（現在検索しているポインタ）を表示する．

:         midptr = (char *)bptr + dsize * dmid;
比較すべきアドレスを算出する．データ列の先頭に（比較地点×要素バイト数）を加えた値である．

:         res = (*compar)(key, midptr);
比較を実行する（結果は，
                  負＝キーの方が小さい
                  0＝等しい
                  正＝キーの方が大きい）

:         if       (res == 0)
:             return    midptr;
比較の結果が等しければ，そのアドレスでデータが見つかったことになる．

:         else if    (res <  0)
:             dmax = dmid - 1;
キーの方が小さければ，検索最大位置を今チェックしたデータの1つ前までとして再び検索を行う．

:         else
:             dmin = dmid + 1;
キーの方が大きければ，検索最小位置を今チェックしたデータの1つ後からとして再び検索を行う．

:     }
:     return    NULL;
ループが終了したということは，データが発見できなかったということである．

: }
```

図9 サンプルプログラム「BSEARCH2」実行画面

```
A>BSEARCH2
bptr = 00A6 dsize = 02 dmid = 03
bptr = 00A6 dsize = 02 dmid = 01
bptr = 00A6 dsize = 02 dmid = 00
00-0000

bptr = 00A6 dsize = 02 dmid = 03
bptr = 00A6 dsize = 02 dmid = 01
bptr = 00A6 dsize = 02 dmid = 00
01-00A6

bptr = 00A6 dsize = 02 dmid = 03
bptr = 00A6 dsize = 02 dmid = 01
02-00A8

bptr = 00A6 dsize = 02 dmid = 03
bptr = 00A6 dsize = 02 dmid = 01
bptr = 00A6 dsize = 02 dmid = 02
03-00AA

bptr = 00A6 dsize = 02 dmid = 03
04-00AC

bptr = 00A6 dsize = 02 dmid = 03
bptr = 00A6 dsize = 02 dmid = 05
bptr = 00A6 dsize = 02 dmid = 04
05-00AE

bptr = 00A6 dsize = 02 dmid = 03
bptr = 00A6 dsize = 02 dmid = 05
06-00B0

bptr = 00A6 dsize = 02 dmid = 03
bptr = 00A6 dsize = 02 dmid = 05
bptr = 00A6 dsize = 02 dmid = 06
07-00B2

bptr = 00A6 dsize = 02 dmid = 03
bptr = 00A6 dsize = 02 dmid = 05
bptr = 00A6 dsize = 02 dmid = 06
08-0000
```

ワンラインNEWS
セイコーエプソンと日本ディジタル・イクイップメントはパソコンの開発・販売など広範囲に業務提携を結んだ．まずパソコンLAN関連商品の開発・販売を行う．

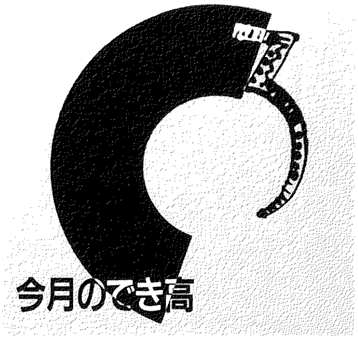
いデータ列範囲を先頭ポインタとして自分自身に引き渡していたが，非再帰版では変数を使用して，データ列範囲の最小ポインタと最大ポイントを管理していく．この2つポインタの動きは，以下のようになる．

①処理開始時には，データ列の先頭をMin，データ列の末尾をMaxとする．
② (Min <= Max) ならば以下の処理を行う．Min > Maxになったら「検索失敗」で処理終了．
③データ列の中央位置を（Min＋Max）／2で算出し，その位置にあるデータと検索対象データを比較する．
④結果が等しければ，「発見」で処理終了．
⑤検索対象データのほうが小さければ，データ列の左半分を対象に再検索となる．Minはそのまま，Maxは③で算出した中央位置の左隣となる．②に戻って検索を続行する．
⑥検索対象データのほうが大きければ，データ列の右半分を対象に再検索となる．Minを③で算出した中央位置の右隣，Maxはそのままとなる．②に戻って検索を続行する．リスト3が非再帰版のBSEARCH2.Cである．

非再帰版のサンプルプログラムBSEARCH2の実行画面を図9に示す．

さきほどと同様に，図10を見ながらバイナリサーチの進行状態を検証してみることにしよう．前述したように，非再帰版では，bptrの値はつねに同じところをポイントさせておき，Max，Minという2つのポインタで中央位置を算出するようにしている．dmidの値は，（Max＋Min）/2　になる．

図10からもわかるように，比較の結果によらずbptrはつねに一定の値になっており，dmidを変化させることによって，次の比較を行う位置を算出するようになる（図11，12）．



それでは，今回のバイナリサーチをテーマツールに応用してみよう．まずクリアしなければならない問題は，説明のところで書いたようにバイナリサーチは「データがソート済みであること」を前提として処理を開始するという点である．現時点でのテーマツールでは，単語リストのデータ並びはファイル内の単語の出現順，つまりランダムになっている（このために，先月号では検索の手法としてシリアルサーチを使用したのだった）．ということは，バ

図10　サンプルプログラム「BSEARCH2」の実行検証

```
bptr = 00A6 dsize = 02 dmid = 03        ※4と比較する  ->  0の方が小さい
bptr = 00A6 dsize = 02 dmid = 01        ※2と比較する  ->  0の方が小さい
bptr = 00A6 dsize = 02 dmid = 00        ※1と比較する  ->  0の方が小さい
00-0000                                 ※0はデータ群に含まれていない

bptr = 00A6 dsize = 02 dmid = 03        ※4と比較する  ->  1の方が小さい
bptr = 00A6 dsize = 02 dmid = 01        ※2と比較する  ->  1の方が小さい
bptr = 00A6 dsize = 02 dmid = 00        ※1と比較する  ->  等しい
01-00A6

bptr = 00A6 dsize = 02 dmid = 03        ※4と比較する  ->  2の方が小さい
bptr = 00A6 dsize = 02 dmid = 01        ※2と比較する  ->  等しい
02-00A8

bptr = 00A6 dsize = 02 dmid = 03        ※4と比較する  ->  3の方が小さい
bptr = 00A6 dsize = 02 dmid = 01        ※2と比較する  ->  3の方が大きい
bptr = 00A6 dsize = 02 dmid = 02        ※3と比較する  ->  等しい
03-00AA

bptr = 00A6 dsize = 02 dmid = 03        ※4と比較する  ->  等しい
04-00AC

bptr = 00A6 dsize = 02 dmid = 03        ※4と比較する  ->  5の方が大きい
bptr = 00A6 dsize = 02 dmid = 05        ※6と比較する  ->  5の方が小さい
bptr = 00A6 dsize = 02 dmid = 04        ※5と比較する  ->  等しい
05-00AE

bptr = 00A6 dsize = 02 dmid = 03        ※4と比較する  ->  6の方が大きい
bptr = 00A6 dsize = 02 dmid = 05        ※6と比較する  ->  等しい
06-00B0

bptr = 00A6 dsize = 02 dmid = 03        ※4と比較する  ->  7の方が大きい
bptr = 00A6 dsize = 02 dmid = 05        ※6と比較する  ->  7の方が大きい
bptr = 00A6 dsize = 02 dmid = 06        ※7と比較する  ->  等しい
07-00B2

bptr = 00A6 dsize = 02 dmid = 03        ※4と比較する  ->  8の方が大きい
bptr = 00A6 dsize = 02 dmid = 05        ※6と比較する  ->  8の方が大きい
bptr = 00A6 dsize = 02 dmid = 06        ※7と比較する  ->  8の方が大きい
08-0000                                 ※8はデータ群に含まれていない
```

図11　比較の結果，検索対象データが小さかった場合のMinとMaxの変化

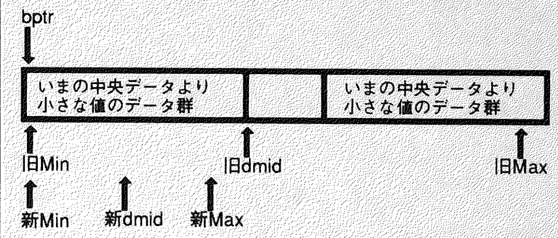
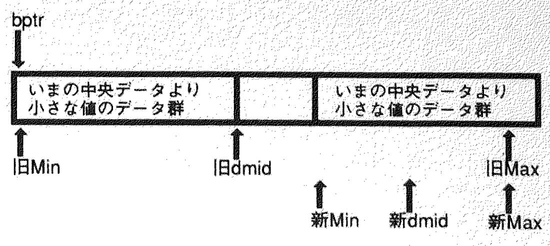


図12　比較の結果，検索対象データが大きかった場合のMinとMaxの変化



ワンラインNEWS
富士通はDTP向けのパソコン「FMR80EP」を発売した．従来機に比べ処理速度を2倍にし，A4判縦表示とA3判横表示の2種類のディスプレイを用意する．

イナリサーチを組み込むとしたら処理を開始する前に単語リストのソートを行わなくてはならないのだろうか．冷静に考えると(別に冷静にならなくともすぐにわかるが)，単語リストをソートしてからバイナリサーチをするのであれば，最初から素直にシリアルサーチをしていったほうが処理は速そうである．では，どうすればよいか…….

ここでバイナリサーチの手順をもう一度思い出していただきたい．データ群の中央付近から検索をはじめて，その位置のデータと検索対象のデータとの大小関係によって，次の検索対象が中央より前か後ろかを決め，検索範囲が1回ごとに半分になっていく．もしデータ群の中に目的のデータがない場合は，検索範囲が1になっても見つからない．ここが注目点である．テーマツールの単語リストの場合，検索結果として「見つからない」ということは，初出の単語ということなので，その検索対象の単語を新たにリストに登録する．その単語を登録すべき場所は，この「最後まで見つからなかった地点」なのである．ここで問題になるのは，検索対象の単語がリスト中に「見つからなかった」時点でのポインタの位置である．話を簡単にするために，単語リストを簡略にしたものでこの点について考えてみよう．

たとえば単語が図13のようなリストになっているときに，「H」を探してみるとする．この場合，検索終了時のポインタは，「J」のところを指している．このとき，最後の比較は「J」と「H」であったはずだが，結果は当然「J」より「H」のほうが小さいということになっているだろう．初出の「H」を登録するには，「J」以降のデータを1つずつ右へ移動させて空いた箇所に「H」をセットする．

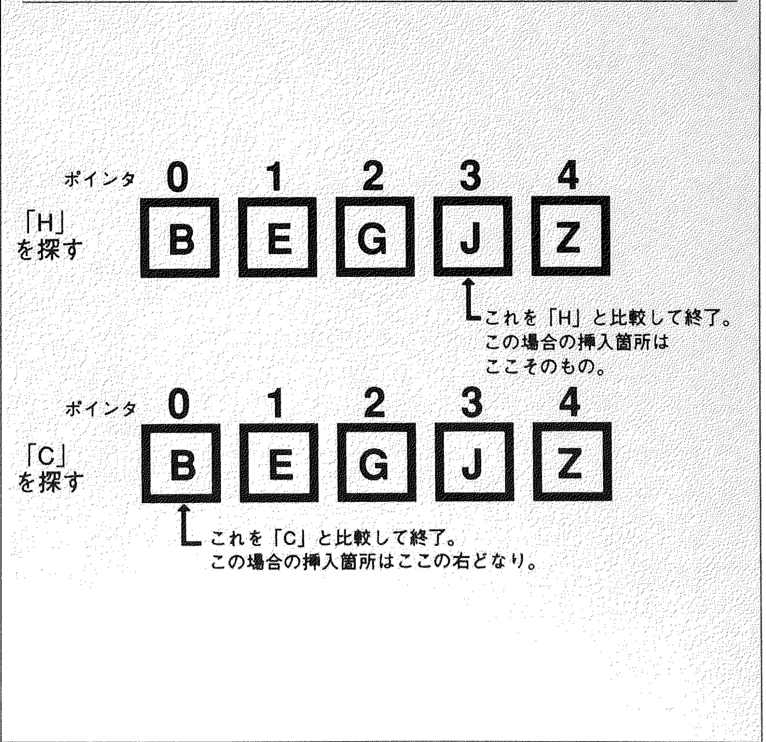
今度は，「C」を検索してみよう．検索終了時のポインタは，「A」のところを指している．最後の「A」と「C」の比較では，「A」より「C」のほうが大きいという結果になっているので，「A」の1つ右隣（この場合は「E」）以降のデータを1つずつ右へ移動させて空いた箇所に「C」をセットする．

つまり，検索対象単語を初出単語としてリストに登録する場合の挿入位置は，検索終了時の最後の比較の結果によって変わってくる．

- **最後の比較によって検索対象単語のほうが小さかった場合**
  →検索終了時に指しているポインタ位置に単語を挿入する．
- **最後の比較によって検索対象単語のほうが大きかった場合**
  →検索終了時に指しているポインタ位置の右隣に単語を挿入する．

テーマツールの単語リスト作成部分を，バイナリサーチを使って，単語リストの中を検索し，見つからなかったときは検索対象単語を初出単語としてリストに登録していくという処理にしたものが，リスト4である．もうお気付きであろうが，この手法で処理をしていくと，終了時の単語リストはすでにソート済みの状態になっている．つまり，いままでのように最後の表示のときにソートを行う必要がないわけだ．いままでせっかく2回にもわたっていろいろなソートを考察してきて，クイックソートによる高速処理を組み込んだのだから少々もったいないような気もするが，テクノロジーの進歩の前にセンチメンタルな感情は無用である．惜し気もなく，このソート部分を削ってしまおう．図14と図15に，新しいWCSUB.Cに含まれる単語「wexist」と「wset」の動作解説をまとめる．

図13　検索終了時のポインタ位置と単語の挿入位置の関係



リスト4 WCSUB.C

```
/* ====================================================================

    WCSUB.C         Copyright 1991,1992 by JUSTECH-LABO. (H.kanoshima)

==================================================================== */
#include    <stdio.h>
#include    <stdlib.h>
#include    <string.h>

#define     WORDMAX     1000

typedef     enum    {BFALSE,BTRUE}  BOOL;

struct  _wlist{
    int     wcount;
    char    *wbody;
}wlist[WORDMAX];

struct  _wlist  *wlptr = wlist;

/* ==========================================================================


========================================================================== */
struct  _wlist *wexist(char *sptr)
{
int     res;
int     dmax;
int     dmid;
int     dmin = 0;

    dmax = wlptr - &wlist[0];
    if(dmax == 0)   return  &wlist[0];
    dmax--;
    while(dmin <= dmax)
    {
        dmid = (dmax + dmin) / 2;

        res = strcmp(sptr, wlist[dmid].wbody);
        if(res == 0)
        {
            wlist[dmid].wcount++;
            return  NULL;
        }
        else if(res < 0)
            dmax = dmid - 1;
        else
            dmin = dmid + 1;
    }
    if(res > 0)     dmid++;
    return  &wlist[dmid];
}

/* ==========================================================================


========================================================================== */
void    wset(char *sptr)
{
char    *cptr;
struct  _wlist  *wlp;

    if((wlp = wexist(sptr)) == NULL)
    {
        putchar('!');
        return;
    }
    if(wlptr >= &wlist[WORDMAX])
    {
        printf("単語数が %d 個を超えました。¥n",WORDMAX);
        exit(1);
    }
    cptr = (char *)malloc(strlen(sptr) + 1);
    if(cptr == NULL)
    {
        printf("メモリが足りないので単語リストが作れません¥n");
        exit(1);
    }
    if(wlptr > wlp)
        memmove(wlp + 1, wlp, (char *)wlptr - (char *)wlp);
    wlp->wbody  = strcpy(cptr , sptr);
    wlp->wcount = 1;
    wlptr++;

    putchar('.');
}

/* ==========================================================================


========================================================================== */

/*  ソートのための比較関数は不要になった。
int     compar(const void *x , const void *y)
{
struct _wlist *xwl = (struct _wlist *)x;
struct _wlist *ywl = (struct _wlist *)y;

    return(strcmp(xwl->wbody , ywl->wbody));
}
*/

/* ==========================================================================


========================================================================== */
void    dump_word(void)
{
struct  _wlist  *wlp;

/* 表示時のソートは不要になった */
/*  puts("¥n*** Sorting ***¥n"); */

/*  qsort(wlist , wlptr - wlist , sizeof(struct _wlist) , compar);  */

        puts("¥n");
    for(wlp = wlist ; wlp < wlptr ; wlp++)
        printf("(%04d) - '%s'¥n" , wlp->wcount , wlp->wbody);
}

/* ==========================================================================


========================================================================== */
char    *wget(char **sptrptr)
{
static  char    getw[2048];
char    *sptrs , *sptrd;
int     cplen;

    sptrs = *sptrptr;

    sptrs += strspn(sptrs , " ,¥t¥n");
    if((*sptrs == '¥0') || (*sptrs == ';')) return  NULL;

    sptrd = strpbrk(sptrs , " ,¥t¥n");
    if(sptrd == NULL)   cplen = strlen(sptrs);
    else                cplen = sptrd - sptrs;
    sptrd = getw;
    while(cplen--)
        *sptrd++ = *sptrs++;
    *sptrd = '¥0';
    *sptrptr = sptrs;

    return  getw;
}
```

ワンラインNEWS
日立製作所は半導体の低電圧化を進める。4ビットMCUでは最小動作電圧が1.8Vで命令実行時間1μ秒の製品を開発。6月からサンプル出荷する。

図 14　バイナリサーチ版単語リスト検索関数wexist

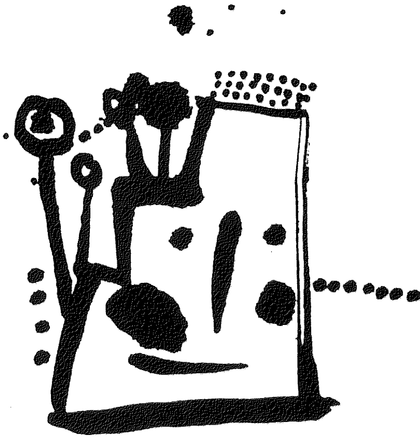
```
: struct    _wlist *wexist(char *sptr)
: {
: int        res;
: int        dmax;
: int        dmid;
: int        dmin = 0;
:
:     dmax = wlptr - &wlist[0];
検索最大位置は，単語リストwlistの要素数

:     if(dmax == 0)    return    &wlist[0];
もし要素数が0ならば，単語リストの先頭ポインタを返す．

:     dmax--;
:     while(dmin <= dmax)
:     {
検索最小位置（初期値0）と検索最大位置が重なる地点まで検索を行う．

:         dmid = (dmax + dmin) / 2;
比較を行う地点は，最小位置と最大位置の中間点である．

:         res = strcmp(sptr, wlist[dmid].wbody);
文字列の比較を実行する（結果は，
                    負＝キーの方が小さい
                    0＝等しい
                    正＝キーの方が大きい）

:         if(res == 0)
:         {
:             wlist[dmid].wcount++;
:             return    NULL;
:         }
文字列が等しければ，出現カウンタを1つアップして，ポインタとしてNULLを返して処理を終了する．

:         else if(res < 0)
:             dmax = dmid - 1;
キーの方が小さければ，検索最大位置を今チェックしたデータの1つ前までとして再び検索を行う．

:         else
:             dmin = dmid + 1;
キーの方が大きければ，検索最小位置を今チェックしたデータの1つ後からとして再び検索を行う．

:     }

:     if(res > 0)        dmid++;
:     return    &wlist[dmid];

: }
```

図15　バイナリサーチ版単語リスト登録関数west

```
: void    wset(char *sptr)
: {
: char    *cptr;
: struct    _wlist    *wlp;
:
:     if((wlp = wexist(sptr)) == NULL)
:     {
:         putch('!');
:         return;
:     }
サーチの結果，NULLが返ってきたら既出単語として処理する．

:     if(wlptr >= &wlist[WORDMAX])
:     {
:         printf("単語数が %d 個を超えました．Yn",WORDMAX);
:         exit(1);
:     }
以降は初出単語の処理である．まず，登録単語数の限界チェックを行い，最大値を越えていたらエラーメッセ
ージを出して終了する．

:     cptr = (char *)malloc(strlen(sptr) + 1);
新しい単語の文字数分だけメモリを確保する．

:     if(cptr == NULL)
:     {
:         printf("メモリが足りないので単語リストが作れませんYn");
:         exit(1);
:     }
メモリ確保に失敗したら，エラーメッセージを出して終了する．

:     if(wlptr > wlp)
:         memmove(wlp + 1, wlp, (char *)wlptr - (char *)wlp);
新規登録すべき位置が，単語リストの最末尾でない場合には，まず挿入位置以降のリストを後ろに1つずらす


:     wlp->wbody  = strcpy(cptr , sptr);
挿入位置に文字列へのポインタをセットする．

:     wlp->wcount = 1;
新規リストとして，カウンタ初期値1をセットする．

:     wlptr++;
単語が1つ増えたので，単語の末尾ポインタを更新する、

:     putch('.');
初出単語の状態表示を行う．

: }
```

先月までのテーマツールは,「単語の出現頻度をリスト化して，最後にそれをソートしながら表示する」という基本構造をもっていたが，今回の改造で「単語の出現頻度をソートしながらリスト化して最後に表示する」というように変更した．この改造の効果は，実際にプログラムを実行してみると一目瞭然である．先月までの改造でもクイックソートやリストの優先順位変更などで効率はアップしていたが，今回の効果はそれらをはるかに凌ぐものである．

さて次回からは，バイナリサーチの手法をさらに発展させていって，データの構造そのものから効率アップの可能性を検討していくことにする．



98活用アセンブラ技法

ブロックデバイス編
第1回

不揮発性RAMDISKドライバの作成

星野 操

今回は，前回までに掲載したデバイスドライバのスケルトンを（多少変更した上で）使用してブロックデバイスドライバを作成することにする．実際に作成するのは，I・OデータのPIO-34シリーズのハードウェアEMSボードを利用したRAMDISKドライバである．

ハードウェアEMSボード用RAMDISK

今回作成したのは簡単なEMSを利用するRAMDISKです．ただ，それだけだとあまりにもおもしろくないので，I・OデータのPIO-34シリーズのハードウェアEMSボードを直接サポートできるようになっています．

現在では，もう仮想86モードを利用したEMSが主流であるといってもかまわないでしょう．EMSボードを利用するとハードウェアの直接的な投資が少なくありません．98の場合，386DXクラスのシステムだとはじめから1MBの拡張メモリがついてくるので，これをEMSとして簡単に利用できます．仮想86モードを利用することによって，多少実行速度の低下が見込まれます．しかし，たとえばFEPをEMSに入れるときなど，EMSの上でプログラムを走らせるときには拡張ボードに実装するハードウェアEMSに比べて，内蔵拡張メモリを利用する仮想EMSは速度的に非常に有利になります．特別な理由がなければ，386システムでは仮想EMSを利用するのが当然ともいえましょう．

リセットで揮発してしまう

ところが，これにも欠点がないわけではありません．拡張メモリはリセットをかけるとメモリチェックによって消去されてしまいます．EMS自体の規格では，リセット時に格納されていたデータを消去しないようにすることができます．EMSのハンドルに名前をつけて，それを用いることでリセット前にアロケートされていたRAMDISKの情報をリセット後に再使用する，ということも可能なのです．しかし，メモリチェックによってデータが消去されてしまっては，どうやってもこのような再使用はかないません．EMSを利用したRAMDISKはリセットのたびにどうしてもデータが揮発してしまうのです．

大きな拡張メモリをもっていても，Windowsでも使用しない限り拡張メモリをそれ自体として扱うことはあまりないので，実際にはそのほとんどをEMSとして割り当てることになるでしょう．そしてEMSのいちばん大きな用途は，結局RAMDISKになってしまうといえましょう．実際にこれを大量に使用するソフトウェアは，一部のカルクがあるぐらいで決して多くはありません（64KBの連続領域を1組だけ使用するというのはよくありますが）．主目的ともいえるRAMDISKがテンポラリにしか使用できないというのは悲しいものがあります．

以前使用されていたバンク切り替えのメモリを利用していたRAMDISKは，当初はリセット時に内容が消去されるものしかありませんでした．しかしそのうちに，メモリの内容をチェックしそれが保存されているようであれば，あえて中身を消去しないというものが主流になっています．たしかにRAMDISKはコンパイラのテンポラリファイル作成など，テンポラリで高速が要求されるときに使用されるのが普通です．しかし，それでも内容が消えるよりは消えないほうが好ましいに決まっています．こういう面で，データの内容が電源を切らない限り保存されるハードウェアEMSなどの拡張スロットにさすボードも捨てがたいものがあるといえるでしょう．

EMMを通さず直接管理

しかし，システムには1つのEMSドライバしかインストールすることはできません．ハードウェアEMSメモリと，仮想86を用いて利用する拡張メモリとを共に1つのEMMで管理することはできます．しかし，その場合揮発する拡張メモリに引きずられて，ハードウェアEMSメモリも内容を保存させることはできなくなります．そこで，ハードウェアEMSのメモリだけをEMMを通さずに管理し，揮発しないRAMDISKの用途で使えるようにしよう，というのが今回のドライバです．通常のEMSのウィンドウのほかに16KBのウィンドウ領域が必要となりますが，リセットによっても保存されるワーク領域というのは，使い方によっては非常に便利なものになるでしょう．

もちろん，ハードウェアを直接操作することになりますから，そのコントロールのための情報を入手しないといけません．今回利用しているI・Oデータのボードのハードウェア的な仕様については，技術評論社の「The BASIC」'90年12月号に掲載されていたので，これを実装してみました．

このほかに，IOCTL呼び出しを利用して，ライトプロテクトがかけられるようにしてあります．実際には，RAMDISKにライトプロテクトなど必要ないでしょうが，これはまあブロックデバイスに対するIOCTLのサンプルというふうにでも考えてください．

ブロックデバイスのデバッグ

ここで簡単にブロックデバイスドライバのデバッグについて触れておきま

しょう．ブロックデバイスの場合は，キャラクタデバイスに比べて，多少デバッグが困難です．手段がまったくないわけではありませんが，基本的にブロックデバイスはキャラクタデバイスとは異なり，adddrvなどのコマンドでデバイスリンクに接続することができません．このため，必ずCONFIG.SYSに記述する必要があります．

ここで，対象となるブロックデバイスが，実際の読み書きはともかく初期化動作だけでも正常にすめば問題は少ないのです．困るのは，初期化さえうまくいかない場合です．うまく立ち上がらないブロックデバイスをCONFIG.SYSに登録すると，システムが立ち上がらなくなり，ひいてはこの原因となっているCONFIG.SYS自身の修正もできなくなってしまいます．

CONFIG.SYSの書き直しと再起動

こうなるとHDだとどうしようもなくなるので，CONFIG.SYSに対象となるデバイスが登録されていないFDから立ち上げないといけません．その後，CONFIG.SYSを修正してから，またHDで立ち上げ，ブロックデバイスのソースを修正したあと，またCONFIG.SYSを元に戻して立ち上げてみる，ということを繰り返さないといけません．

しかし，メモリスイッチによってブートアップドライブがSASIやSCSIのHDに限定されていると，FDからの立ち上げはできなくなってしまいます．このような場合，まずメモリスイッチをディップスイッチなどで1度初期化してからFDで立ち上げ，その後にメモリスイッチの設定値をもとに戻すということになります．これを毎回行っていてはたまりません．かといって，ブートシステムを「標準」にしていると，HDの起動メニューが出てきてしまうなど不都合が少なくありません．

これを避けるためにいくつかの手段をとる必要があります．1つは，ブート装置は標準に設定したままでも自動的に立ち上がるようにしておく方法です．最近のHD起動プログラムだと，起動可能領域が1つしかないと自動的にその領域から立ち上がるようになっているようです．ですから，起動可能領域を制限しておくという手があります．複数のDOSバージョンから立ち上げたい場合など，この手が使えない場合には，HDの起動メニューを自動的にカットする（FDがあればFDから，そうでなければHDのデフォルトの起動領域から立ち上げる）方法があります．これを行うプログラムは，ネットワークなどで配布されています．

ただ，前記のような方法だとFDで起動したときの環境は一般にHDで起動したときと異なる（少なくとも，ドライブ名のつけ方が変わってくる）ので，そのままデバイスソースを修正してアセンブルをしたりするのにはあまり向きません．一般的には，CONFIG.SYSだけを修正し，またHDで立ち上げることになります．これを避けるためには，たとえば複数のCONFIG.SYSセットを用意し，これを起動時に選択できるようにしておく方法があります．そのためのプログラムは，やはりネットワークなどで配布されています．

筆者がよく使用する方法は，デバイスドライバだけをFDにおく，というものです．こうした場合に，普通にFDを入れておけばそのドライバがロードされますし，システムを立ち上げないような不正なドライバであった場合に

ワンラインNEWS
キヤノンはRISC型MPUを搭載し，76MIPSの処理能力を持つオフコン「システム3000シリーズ900」を発売した．同シリーズは米ヒューレット・パッカードからのOEM.

はFDを抜いておけば，そのデバイスの組み込みはスキップされます．この場合，起動装置はもちろんHDに指定しておいてかまいません．

以上のような対策をとった上で，とにもかくにも組み込みだけは正常にできるようにしないといけません．そうしないとあとの話が続かなくなります．どうしても正常に組み込めない場合には，組み込んだあと実際に使用するときには必要でも，さしあたっては必要ない初期化ルーチンなど，どんどん外していきます．初期化のときには，DOSの出力命令などを使用することができますから，適当な部分にプローブを入れて，どこまで初期化が進行したかをチェックしながら状態を調整することにします．

立ち上がってからのデバッグ

そして，まがりなりにも組み込めるようになったら，あとは徐々に初期化ルーチンを要求される状態に復元していきます．1度立ち上がってしまえば，あとはデバッガなどが使用できるので，適当な領域を用意してそこに動作のログを記録し，それをあとから検索することによって動作の状況を確かめることができます．たとえば，今回のブロックデバイスの場合は，マッピングするページ番号を特定の領域に格納しておくことで，マッピングアルゴリズムが正常に動作しているかどうかをチェックしました．

ブロックデバイスの場合，初期化部分，つまりinitファンクションと同様に，BUILD BPBファンクションも正常に動作していることを確認しないといけません．これを見るためには，ASCIIのTOOLSにあったDISKSTATのようなDPB（BPBより作成される情報領域）を参照するようなプログラムでBPBの値が適切であるかどうかを判断すればよいでしょう．

その後は，通常の読み書きが正常にできるかどうかを調べます．DOSのCOPYコマンドでファイルを書き込んでみてもいいのですが，この場合ディレクトリなどの内容はシステムバッファに残っているので，ストップキーを押してバッファの内容をフラッシュしてからチェックする必要があります．

しかし，セクタの読み書きを直接チェックするには，SYMDEBなどのL/Wコマンドで直接セクタ読み書きを実行してみるほうが安全でしょう．前記のように読み書きの進行状態のログをメモリに書き込んでおけば，L/Wコマンド実行後にそのメモリの内容を調べることで，動作状況をチェックすることができます．

以上のような作業を繰り返してデバッグを行うことになります．

今回のプログラム xdrive.sys

それでは，実際のプログラムについて説明しましょう．

リスト1　DEVSTD.ASMへの変更部分

```
-------------------------------------------------------
code    segment public
        assume  cs:code, ds:code

bios_id equu    <'X$SCSI$X'>

; device header
device_header:
ifdef   PRIME_DEV
        dd      -1                      ; link pointer
        dw      0c000h                  ; character device, ioctl
        dw      strategy
        dw      interrupt
        db      bios_id
elseifdef       CHAR_DEV
        dd      -1                      ; link pointer
        dw      8000h+DEVINFO           ; character device, ioctl
        dw      strategy
        dw      interrupt
drive_number    label   byte
        db      DEVNAME,
else
        dd      -1
        dw      0+DEVINFO
        dw      strategy
        dw      interrupt
drive_number    db      ?
endif
-------------------------------------------------------
ifdef           BLOCK_DEV
        include xdrive.asm
elseifdef       CHAR_DEV
        include xtape.asm
else    ;       PRIME_DEV
        include xscsi.asm
endif
-------------------------------------------------------
init:
        ; message output
        mov     dx,offset scsi_mes
        mov     ah,9
        int     21h

ifndef  PRIME_DEV
ifndef  STAND_ALONE
        ; search device
        call    search_bios
        @if     c
                ; message output
                mov     dx,offset not_bios_mes
                mov     ah,9
                int     21h

                ; never install
                mov     word ptr ds:[4],0
                les     bx,packet
                mov     word ptr es:[bx].trans,0
                mov     word ptr es:[bx].trans+2,cs
                mov     drive_number,0
                mov     es:[bx].media,0
                mov     ax,100h
                jmp     interrupt_end
        @endif
endif
endif

        ; option parser invoke
        call    parse_option

        call    ex_init
        @if     c
                ; never install
                mov     word ptr ds:[4],0
                les     bx,packet
                mov     word ptr es:[bx].trans,0
                mov     word ptr es:[bx].trans+2,cs
                mov     drive_number,0
                mov     es:[bx].media,0
                mov     ax,100h
                jmp     interrupt_end
        @endif

        call    check_option
-------------------------------------------------------
```

ワンラインNEWS
メルコは日本電気のパソコン新機種に対応したRAMボードなどを発売した．ノート型パソコン用RAMカードに，まだサンプル出荷が主流の16MビットのDRAMを使用．

DEVSTD.ASMの変更

さて，まず最初に以前掲載したDEVSTD. ASMを多少変更します．変更の要点は，デバイスの作成にあたってより汎用性を高めるためのものです．変更点は3つのブロックに分かれており，最初のデバイスヘッダの部分，それから実際のデバイスのインプリメンテーションをインクルードする部分，そして最後の初期ルーチンの部分です．

デバイスヘッダは，まずIFDEFでTAPE_DEVの定義の有無を要求していた部分をCHAR_DEVに変更しています．これは，より一般的にキャラクタデバイスとして使用できるようにということです．そしてデバイスインフォメーションワードとして設定する値を定数の直接記述ではなく，数値マクロを利用して行っています．これは，ドライバのインプリメンテーション部分でインフォメーションワードを定義できるようにするためです．以前のバージョンでは，ブロックデバイスはIOCTLが受け取れない形になっていましたが，このような部分を容易に変更できるようになります．

また，キャラクタデバイスを使用する場合，そのデバイス名を外部から指定できるようにしてあります．ここでMASMのように2パスのアセンブラであれば，1パスめはダミーのデバイス名(たとえば,空白8文字)を指定し，2パス目でインプリメンテーション部分で定義した文字マクロを用いて実際のデバイス名を定義することも可能です．

しかし,前のTASMのように1パスアセンブラであれば，このような処理をすることはできません．ですから起動時に,/Dオプションで直接デバイス名を定義してやることにしないといけませんでした．このとき，マクロの「内容」に空白を含めることはできないので，定義された名前の後ろにダミーの空白を定義しておきます．これによって，空白がつめられた8文字のデバイス名を保証しています．

ソース中央での変更部分というのは，デバイスヘッダ定義部分に合わせてTAPE_DEVをCHAR_DEVに変更しているだけです．ここでインクルードするファイル名自体は変更されていないので，別のキャラクタデバイスを作成する場合には，ファイル名はXTAPE. ASMである必要があります．最後の変更部分は，まずSCSIコントロールデバイスを必要としないデバイスドライバを記述できるようにするために，デバイス存在チェックルーチン部分をもう一重IFNDEFで括ってあります．シンボルSTAND_ALONEが定義されている場合には，このコントロールデバイスのチェックルーチンはアセンブルされません．さらに，ex_init呼び出し直後に，キャリーフラグをチェックし，キャリーが立っていた場合には組み込みを中止するようにしてあります．これによって，インプリメント部分でなにか不正な状態を発見した場合には組み込みを中止することができるようになりました．

XSCSI／XTAPEの変更

この新しいDEVSTDと以前のXSCSI. ASM／XTAPE. ASMを組み合わせて使用する場合には，XSCSI／XTAPEに次のような変更を加える必要があります．

まず，XSCSIのほうはデバイスインフォメーションの定義は不要なので，ex_initの部分だけ対応すれば十分です．このプロシージャのret命令の前にAND AX, AXでキャリーをクリアするようにすればOKです．

一方XTAPEのほうは，同様にex_initの処理をするほかに，DEVIN-

ワンラインNEWS
米テキサス・インスツルメンツは高集積化を実現したカラーパレットICを開発，日本市場でサンプル出荷を開始した．従来製品に比べ周辺回路を内蔵しチップ数を削減

リスト2　xdrive.asm

```
;----------------------------
; xdrive: I/O data EMS access driver
;----------------------------

;----------------------------
; constants and works
;
MY_MEDIA        equ     0a5h                    ; my media descripter byte
DEVINFO equ     4000h                           ; accept ioctl

partial_page_map_struct dw      1               ; used to save/restore page map
ems_frame       dw      0                       ; frame address
emm_handle      dw      -1
last_page       dw      -1
write_protect_flag      db      0               ; write protect flag


;----------------------------
; invalid entries
;
error_command:
nd_read:
read_status:
read_flush:
write_status:
write_flush:
ioctl_in:
removable_media:
output_until_busy:
device_open:
device_close:
deinstall:
        mov     ax,e_bad_command
        jmp     interrupt_end


;----------------------------
; exit without error
;
done:
        mov     ax,100h
        jmp     interrupt_end

;----------------------------
; media check, always return NOT CHANGED
;
media_check:
        les     bx,packet
        mov     word ptr es:[bx].trans,1        ; not change flag
        jmp     done

;----------------------------
; build BPB, return predefined BPB structure
;
build_bpb:
        les     bx,packet
        mov     al,media_descriptor_byte
        mov     es:[bx].media,al
        mov     es:[bx].count,offset bpb
        mov     es:[bx].start,cs
        jmp     done

;----------------------------
; BPB structure
;
bpb     label   byte
sector_size     label   word
        dw      1024                            ; sector size
        db      1                               ; sector per cluster(CONST)
reserved_sector label   word
        dw      0                               ; reserved sectors
        db      1                               ; fat number
directory_entry label   word
        dw      128                             ; directory entry
total_sector    label   word
        dw      2048                            ; total sector count
media_descriptor_byte   label   byte
        db      MY_MEDIA                        ; media descriptor byte
sector_per_fat  label   word
        dw      3                               ; sector number per fat

shift_factor    db      ?                       ; sector size shift factor
mask_factor     dw      ?                       ; lower mask data

;----------------------------
; BPB entry list, used in init function
;
bpb_array       label   word
        dw      bpb


;----------------------------
; ioctl function
;
ioctl_out:
        les     bx,packet
        les     bx,es:[bx].trans
        mov     al,es:[bx]
        mov     write_protect_flag,al
        jmp     done


;----------------------------
; map memory into window
; input:
;       DX: sector number
; may break AX, CX
; hold DX, SI, DI

map_ems proc
        cmp     cs:emm_handle,-1
        @if     nz
                ; use standard ems
                push    dx              ; save dx reg
                mov     ax,4400h        ; set function code
                mov     bx,dx           ; calculate logical page
                mov     cl,14           ; get shift count
                sub     cl,cs:shift_factor
                shr     bx,cl
                cmp     bx,cs:last_page
                @if     nz
                        mov     cs:last_page,bx
                        mov     dx,cs:emm_handle
                        int     67h     ; invoke emm
                @endif
                pop     dx
        @else
                mov     al,0            ; assume window number:0
                out     0e8h,al         ; output to port
                mov     al,byte ptr cs:ems_frame+1      ; set window address
                or      al,1            ; set window enable flag
                out     0e9h,al         ; output to port

                mov     ax,dx           ; convert sector number to page
                mov     cl,14           ; shift factor of window width
                sub     cl,cs:shift_factor      ; get shift count
                shr     ax,cl
                out     0eah,ax         ; set page number to port
        @endif

        ret
map_ems endp


;----------------------------
; save partial page map
;
page_map_contents       db      32 dup (?)

save_map        proc
        ; check using standard EMS
        cmp     emm_handle,-1
        @if     nz
                mov     si,offset partial_page_map_struct
                push    cs
                pop     es
                mov     di,offset page_map_contents
                mov     ax,4f00h
                int     67h
        @endif

        ret
save_map        endp


;----------------------------
; restore saved page map
;
restore_map     proc
        ; check using standard EMS
        cmp     emm_handle,-1
        @if     nz
                ; restore
                mov     si,offset page_map_contents
                mov     ax,4f01h
                int     67h
        @endif

        ret
restore_map     endp


;----------------------------
; read function entry
;
read:
        call    save_map
        les     bx,packet               ; get parameters
        mov     cx,es:[bx].count        ; cx: transfer count
        mov     dx,es:[bx].start        ; dx: sector number
        les     di,es:[bx].trans        ; es:di transfer address

        call    read_sub
        call    restore_map
        jmp     done


;----------------------------
; read function body
;
read_sub        proc
        mov     ds,cs:ems_frame         ; get ems window segment

        @begin
                or      cx,cx           ; loop until cx=0
        @while  nz
                push    cx
                call    map_ems         ; show ems page memory
                mov     si,dx           ; get source offset in the window
                mov     cl,cs:shift_factor
                shl     si,cl
                and     si,3fffh        ; mask higher bit

                mov     cx,cs:sector_size
                shr     cx,1
                rep     movsw           ; transfer

```

FO=4800Hとしてデバイスインフォメーションの定義をしなければいけません．さらにアセンブル時のオプションを変更します．以前の場合は「/DTAPE_DEV」としていましたが，今回は「/DCHAR_DEV /DDEVNAME='XTAPE'」と2つのマクロを定義してやることになります．

DEVINFOの定義

では，ブロックデバイスのインプリメンテーション部分，XDRIVE.ASMについて説明しましょう．最初に，テープデバイスと同様にDEVINFOの定義があります．これは，IOCTLを受け付けるブロックデバイスという指定です．通常，ブロックデバイスはIOCTLを受け付けるようには書かれませんが，このようにデバイスインフォメーションのビットをセットすれば，IOCTL文字列を収受することも可能です．このほかに，ドライブのメディアディスクリプタバイトの値が定義されています．これは実際にはどんな値でもかまいません．

EMSアクセス関係の変数領域

それからEMSアクセス関係の変数領域があります．ライトプロテクトフラグもここに置いておきます．emm_frameは0で初期化してありますが，ここにウィンドウのセグメント値が入ります．この値が起動時オプションで設定された場合には，I・OデータのハードウェアEMSを指定されたウィンドウアドレスで直接コントロールすることを意味することにします．これが変更されず0のままであれば，通常のEMMを利用してRAMDISKを構成することにします．

emm_handle

またemm_handleはEMMで使用されるハンドルを格納するものであり，ハードウェアEMSを使用しているかどうかについては，この値が-1のままであるかどうかによって判断することができます．last_pageはEMMを利用する場合に使うもので，ここにマップされたページ番号を記録することで，同一のページを再マップするのにEMMを複数回呼び出して無駄な時間を使うのを避けます．

なお，EMSをRAMDISK目的でアクセスする場合には，以前のページ状態を保存し，アクセス後に復元しないといけません．このためにはEMMハンドルをIDとして使用することができますが，万が一にもEMM内部で用意されているワークエリアを使い果たすことがあると，ディスクのアクセスエラーになってしまいます．これを避けるために，EMMのパーシャルページマップのセーブ・リストアというファンクションを利用しています．

このファンクションを利用するためには，セーブするウィンドウのセグメントを配列にしたものを渡してやる必要があります．ここでは変数emm_frameを二重に使用し，この配列の意味ももたせてやります．その直前に定義されているpartial_page_map_structは配列の要素数を示しているものであり，これら2つの変数の相対位置を変更することがあってはなりません．

その次に，サポートされていないファンクションの処理部分があり，これらはエラーコードをもって復帰することにしてあります．次のdoneは正常終了部分です．media_checkはドライブのメディアが交換されたかどうかをチ

ワンラインNEWS
ジャストシステムは「大地PROシステム30」専用の「大地VAF・SAPLSファイルコンバータ」」を発売．大地の文書を電算写植機「SAPLS」シリーズに変換できる．

```
			pop	cx			; resume count
			inc	dx			; advance sector pointer
			dec	cx
		@repeat

		push	dx
		mov	dx,-1				; unmap ems
		call	map_ems
		pop	dx
		push	cs
		pop	ds
		ret
read_sub	endp


;----------------------------
; write function entry

write:
write_verify:
		; check protect flag
		cmp	write_protect_flag,0
		@if	nz
			; protected
			mov	ax,e_write_protect
			jmp	interrupt_end
		@endif

		call	save_map
		les	bx,packet		; get parameters
		mov	cx,es:[bx].count	; same as read function
		mov	dx,es:[bx].start
		lds	si,es:[bx].trans

		call	write_sub
		call	restore_map
		jmp	done


;----------------------------
; write function body

write_sub	proc
		mov	es,cs:ems_frame		; get ems frame segment

		@begin
			or	cx,cx		; until tranfser count = 0
		@while	nz
			push	cx		; just same as read function body
			call	map_ems		; except that transfer direction is
			mov	di,dx		; reverse
			mov	cl,cs:shift_factor
			shl	di,cl
			and	di,3fffh	; mask higher bit

			mov	cx,cs:sector_size
			shr	cx,1
			rep	movsw

			pop	cx
			inc	dx
			dec	cx
		@repeat

		push	dx
		mov	dx,-1				; unmap ems
		call	map_ems
		pop	dx
		push	cs
		pop	ds
		ret
write_sub	endp

;----------------------------
; from here, not resident part

end_label:


;----------------------------
; bootup message

scsi_mes	db	'EMS ramdisk driver',0dh,0ah,'$'
clear_mes	db	'Drive data cleared',0dh,0ah,'$'
non_emm_mes	db	'EMM not installed',0dh,0ah,'$'
shortage_mes	db	'Free pages not left',0dh,0ah,'$'

emm_id	db	'EMMXXXX0'


;----------------------------
; startup sector buffer

sect_buf	db	2048 dup (0)


;----------------------------
; initialize routine

ex_init proc	near
		; parse option
		push	cs
		pop	es
		mov	bx,offset options
		@begin
			; get argument value
			push	bx
			push	es
			les	bx,[bx].o_arg
			call	atoh
			pop	es
			pop	bx

			mov	dl,[bx].o_char
			cmp	dl,'w'
			@if	z
				mov	[bx].o_char,0
				mov	ems_frame,ax
			@else
			cmp	dl,'s'
			@e_if	z
				mov	[bx].o_char,0
				mov	total_sector,ax
			@endif

			add	bx,5
			cmp	bx,offset options_tail
		@until	nc

		; calculate BPB parameters
		mov	ax,total_sector		; must be 12bit fat
		sub	ax,reserved_sector
		mov	bx,ax
		mov	ax,directory_entry	; calc sector number for directory
		mov	cl,5			; *32
		shl	ax,cl
		add	ax,sector_size		; check odd byte
		dec	ax
		mov	dx,0
		div	sector_size		; ax is sector number for dir
		sub	bx,ax
		mov	ax,bx			; *1.5
		inc	bx
		shr	bx,1
		add	ax,bx
		add	ax,sector_size		; calculate sector number for fat
		dec	ax
		mov	dx,0
		div	sector_size
		mov	sector_per_fat,ax

		; calculate shift factor
		mov	shift_factor,0
		mov	mask_factor,0
		mov	ax,sector_size
		@begin
			cmp	ax,1
		@while	nz
			shr	ax,1
			inc	shift_factor
			stc
			rcl	mask_factor,1
		@repeat

		; check use of standard ems
		cmp	ems_frame,0
		@if	z
			; check existense of emm
			mov	ax,0		; get emm id string address
			mov	es,ax
			mov	es,es:[67h*4+2]	; get emm segment
			mov	di,0ah		; offset = 0ah
			mov	si,offset emm_id
			mov	cx,8		; id length
			repz	cmpsb
			@if	nz
				; not ems installed
				mov	dx,offset non_emm_mes
				mov	ah,9
				int	21h
				stc		; error flag
				ret
			@endif

			; decide window address
			mov	ah,41h
			int	67h
			mov	ems_frame,bx	; bx is window base

			; allocate memory
			mov	ax,total_sector	; get required ems page number
			mov	bx,ax
			mov	cl,14		; calculate shift number
			sub	cl,shift_factor
			shr	bx,cl
			test	ax,mask_factor	; check odd sectors
			@if	nz
				inc	bx
			@endif

			mov	ah,43h		; allocate
			int	67h

			; check error
			or	ax,ax
			@if	nz
				; error, may be shortage memory
				mov	dx,offset shortage_mes
				mov	ah,9
				int	21h
				stc		; error flag
				ret
			@endif

			mov	emm_handle,dx
		@endif

		; read 1st sector
		mov	cx,1
		mov	dx,reserved_sector
```

ワンラインNEWS
米アップル・コンピュータの日本法人アップル・コンピュータ・ジャパンは2月10日からアップル・コンピュータに社名変更した。本社も渋谷区千駄ヶ谷に移転する。

```
		push	cs
		pop	es
		mov	di,offset sect_buf
		call	read_sub

		; check contents
		@mc
			mov	al,media_descriptor_byte
			cmp	sect_buf,al
			@cond	nz
		@or
			cmp	word ptr sect_buf+1,0ffffh
			@cond	nz
		@or
			; check key shift/ctrl
			mov	ah,2
			int	18h
			and	al,11h
			cmp	al,11h
			@cond	z
		@if
			; output message
			mov	dx,offset clear_mes
			mov	ah,9
			int	21h

			; clear buffer
			push	cs
			pop	es
			mov	di,offset sect_buf
			mov	cx,(size sect_buf)/2
			mov	ax,0
			rep	stosw

			call	save_map
			mov	dx,reserved_sector	; from 1st sector
			@begin
				mov	cx,1	; write buffer contents
				mov	si,offset sect_buf
				call	write_sub
				cmp	dx,30	; till arbitral number of sector
			@until	nb

			; and write fat
			mov	al,media_descriptor_byte
			mov	sect_buf,al
			mov	word ptr sect_buf+1,0ffffh
			mov	dx,reserved_sector
			mov	cx,1
			mov	si,offset sect_buf
			call	write_sub

			call	restore_map
		@endif

		mov	drive_number,1

		and	ax,ax			; clear error flag
		ret
ex_init endp
```

ェックするものであり，ここでは常に交換されていないということで1を返しています．

build_bpb

build_bpbはBPBの内容を返すためのものであり，その後に用意されているBPB構造体のアドレスをパケットに設定して復帰します．BPBは直接構造体を用いては定義していませんが，その内容はコメントしてあるので問題はないでしょう．ここでセクタサイズは一応変更可能なようにしてあるはずですが，1クラスタあたりのセクタ数は1で変更できません．ディレクトリエントリの数も変更可能です．このほか，トータルセクタ数はオプションで設定可能となっており，FATあたりのセクタ数などは起動時に計算して設定するようになっています．その後ろにあるshift_factorおよびmask_factorはセクタ長に依存して起動時に設定されます．bpb_arrayはinit実行時にそのアドレスが渡されるもので，ここでは1つのbpbへのポインタを要素としてもっているだけです．

ioctl_out

ioctl_outはここではライトプロテクトのフラグを設定するために使用しています．バッファのアドレスを取得し，その先頭の値をそのままフラグに設定します．これが非0になると，書き込みを拒否するようにします．なお，ioctlには入力と出力の2つのファンクションが，キャラクタデバイス・ブロックデバイスそれぞれに用意されています．ただし，レジスタを経由して受け渡しするのと異なりバッファのアドレスと長さだけが指定されるので，入出力の間には実質的な相違はありません．単にエントリポイントが異なっているだけということができます．だから，極端な話ioctl_outで入力をすることも可能です．

ですから，ioctlのモードを示すサブコードの値はかなり曖昧です．ここでは，ブロックデバイスにioctl文字列を送るのに，サブコード5を使用して，デバイスドライバのほうではioctl_outで受けています．しかしこれはたとえば，アスキー出版のいわゆる「黒本」とは入出力の表現が逆転しています．入出力の定義はデバイスドライバとクライアントプログラムのどちらから見るかで逆転してしまうので厄介なのですが，どうも黒本の表現ではなく今回使用したもののほうが正しいようです．

map_ems

次のmap_emsはウィンドウにEMSメモリをマップするために使用します．まずemm_handleをチェックすることでEMMを利用するか直接ハードウェアをアクセスするかを決定します．DXレジスタにはセクタ番号が入っているので，これを次にページ番号に変換します．実際には，シフトすることになるわけですが，そのビット数はshift_factorを参照することによって決定します．セクタサイズが異なると，シフトするビット数も異なってくるからです．

EMMを利用する場合には，last_pageと比較して無駄なマッピングを防ぎます．ハードウェアを直接アクセスする場合には，記述された手順でI/Oに出力を行います．特定のページをメモリ空間に出すためには，基本的に1つのワードout命令で足りることに

ワンラインNEWS
リコーはパソコン販売体制を強化する．日本IBMからOEM調達している「PS/55ノート」や自社独自仕様の「Mr.マイツール」シリーズのサポート拠点を拡充する．

なります．

なお，この関数を呼び出すときにはDSレジスタが保存されている保証がありません．ですから変数に対するアクセスにはCSプレフィクスが必要となります．今回は，最初のemm_handleの参照にプレフィクスが不足しているバグを取るのに苦労しました．こういう部分では，プレフィクスを明示的に指定せずに，assume疑似命令でDS:nothingと指定して，CSプレフィクスを自動的に発生させるほうがよいかもしれません．

save_map，restore_map

次のsave_mapとrestore_mapは，EMMを利用しているときに，初期のページマップ状態を保存・復元するものです．先に述べたウィンドウアドレスの配列を引数に取ります．保存時の状態保存用ワークエリアは一応32バイト確保してあります．これのEMMファンクションがどれだけの領域を必要とするかは，実はEMMのインプリメントによって確定しません．そのサイズを取得するファンクションもありますが，ダイナミックなメモリアロケーションをする場合(スタック領域を使用する場合も含む)以外は，まず役に立ちません．実際には，大きくても5バイト程度しか必要とはされないでしょう．ここでは，十分に余裕をとっているつもりです．

read

その後のreadは読み込みのファンクションエントリです．
ここではsave_mapを実行してから，レジスタにパラメータを設定してread_subを呼び出しています．そして最後にrestore_mapを実行して終了します．writeエントリも同様です．これは初期化ルーチンの中でセクタの読み書きを実行する必要があるので，その時にパラメータをレジスタ渡しできたほうが好都合だからです．

read_subではまずレジスタ番号に対応したEMSページをマップした後，ページ番号の下位ビットからページ内のオフセットアドレスを計算します．このときのシフトビット数もshift_factorを用います．それからrepmovswでデータの転送を行います．最後にセクタ番号の更新を行い，指定されたセクタ数だけループを繰り返します．

ループを終了したら，ページ番号-1でmap_emmを呼び出し，ウィンドウをインアクティブにします．それから壊されているDSレジスタをここで復元して，復帰します．

write

writeについてもほとんど同様ですが，ライトプロテクトチェックが追加されています．このフラグが立っていた場合には，ライトプロテクトエラーのエラーコードをAXに設定し，直接復帰します．

以上の部分がドライバの常駐時にメモリに残される部分であり，end_label以降の部分は初期化後切り放されます．非常駐部分ではまず，起動時のメッセージ文字列が定義されています．emm_idはEMMの存在チェックに利用します．sect_bufは，初期化時に利用する1セクタ分のバッファ領域です．

ex_init

初期化ルーチンex_initでは，まず起動時のオプション解析をします．wオプションはハードウェアEMSボードのウィンドウアドレスを指定するものです．これが指定されない場合には，通常のEMMを使用することを示します．もう1つsオプションを与えることが可能で，これによって総セクタ数(1KB/セクタならKB単位)を16進で指定します．これが指定されない場合は，デフォルトの2048セクタが有効になります．

オプション解析が終了した後，BPBパラメータの計算をします．総セクタ数から予約セクタ・ディレクトリ用のセクタを引き，12ビットFATとしてFAT用のセクタ数を計算します．ここではほんとうならさらにFATセクタの分を引いて計算し直さないといけなくなるのですが，実際問題としてFAT用のセクタが1セクタ多くても影響はないでしょうから再計算はしていません．次にセクタサイズからshift_factorを計算します．

wオプションが指定されていない場合には，その後EMSのメモリの確保などを行わないといけません．まず最初にEMMがインストールされているかどうかをチェックします．EMMが存在していない場合には，メッセージを出力した後キャリーをセットして復帰します．これによって，組み込みが中止されることになります．EMMの存在チェックは，int67Hの割り込みエントリのセグメントのオフセット0AHから，EMMXXXX0という文字列が存在しているかどうかによって判断されます．

次に，ウィンドウアドレスを取得します．このドライバではウィンドウ0を使用するので，ウィンドウベースがそのままウィンドウアドレスとして使用できます．そして最後に必要なだけ

のページ数を計算し，確保します．もし，確保に失敗したらやはりキャリーをセットして終了し，組み込みを中止します．ここでは，もちろん可能なだけのメモリを割り当てるように変更することもできます．

以上の処理がすんだ後，最初のセクタ，ここではFATの先頭セクタを読み込みます．そしてその先頭部分が不正な値であるか，あるいは組み込み時に[shift]と[ctrl]が共に押されていればフォーマット動作を実行します．フォーマットは，最初にクリアされた内容を適当なセクタ数だけ書き込み，最後にFATの先頭部分にデスクリプタバイトなどを書き込むことによって実行します．ここでも一応，EMSページの内容のセーブ・リストアは実行しています．最後にドライブ数を設定してex_initを終了します．

プログラムの作成

これからxdrive.sysを作成するには，次のような手順を踏みます．

```
tasm /DBLOCK_DEV /DSTAND_ALONE devstd, xdrive
tlink /t xdrive,xdrive.sys
```

なお，ライトプロテクト状態をコントロールするプログラムを別に示します(リスト3)．これは，ドライブ番号とON／OFFの状態とを引数に取り，それに応じてIOCTLを発行します．プログラムは，最初にドライブ番号のレンジをチェックした後，モードの引数をチェックします．もちろん，ここで正しいドライブ番号を指定する必要があります．またモードチェックについては，引数の先頭2バイトをワードコンペアしてチェックしているだけです．比較すべき値は，直接”で括って定義してあります．このとき，下位バイトと上位バイトがひっくり返る問題があるので，onという文字列をチェックするためには，定数'no'と比較を実行することになります．

このプログラムは，単にアセンブル・リンクするだけで実行ファイルが作成できます．これでライトプロテクトをONにした場合には，RAMDISKの書き込みにおいてエラーが発生するようになります．

参考文献

技術評論社　TheBASIC　90/12「アイオーデータ機器PIO-PC34シリーズのアーキテクチャ」小高輝真

リスト3　wprotect.asm

```
;---------------------------
; wprotect: set write protect program
;
;---------------------------

        dosseg
        .model  small,pascal
        masm51

        include condx.inc
        include utils.inc

        .stack  2000h

        .data
iobuf   db      ?

;---------------------------
; initialize routine

        .code
entry:
        @setup
        push    _argc
        push    _argv
        call    _main
exit:
        dos     <4ch,,,>

;--------------------------
func    main    <argc, argv>

        ; check argument
        cmp     argc,3
        @if     nz
                @string <'usage:wprotect drive (ON|OFF)',0dh,0ah>
                @puts   dx
                mov     al,1
                ret
        @endif

        ; get drive name
        mov     si,argv
        mov     di,si[2]
        mov     bl,[di]
        mov     bh,0
        or      bl,20h
        sub     bl,'a'-1
        @mc
                cmp     bl,1
                @cond   b
        @or
                cmp     bl,26
                @cond   a
        @if
                @string <'bad drive name',0dh,0ah>
                @puts   dx
                mov     al,1
                ret
        @endif

        ; get flag
        mov     di,si[4]
        mov     ax,[di]
        or      ax,2020h
        cmp     ax,'no'
        @if     z
                mov     ax,1
        @else
        cmp     ax,'fo'
        @e_if   z
                mov     ax,0
        @else
                @string <'bad argument',0dh,0ah>
                @puts   dx
                mov     al,1
                ret
        @endif

        ; send ioctl
        mov     iobuf,al
        mov     dx,offset iobuf
        mov     cx,1
        mov     ax,4405h
        int     21h

        @if     c
                @string <'ioctl error',0dh,0ah>
                @puts   dx
                mov     al,2
                ret
        @endif

        @string <'Done',0dh,0ah>
        @puts   dx
        mov     al,0
        ret
endf

        end     entry
```


ワンラインNEWS
日本シリコングラフィックスは3次元グラフィックWSの新シリーズ「IRIS Crimson」を売り出す．米ミップス社のRISC型MPU「R4000」を初めて搭載した．

SE-NET拾い読み

ひさびさの「SE-NET拾い読み」です．最近は，「ドラゴンポスト［Free talk］」ボードなどに人気があり，「プログラマ内輪話」ボードは多少静かなようです．それでも，ソースとドキュメントを同時に作成するためのツールの話題などが取り上げられました．

ここでは，SE-NETの「プログラマ内輪話」ボードと「ドラゴンポスト［Free talk］」ボードから，いくつかの楽しい書き込みをご紹介します．また，書き込みについては編集の都合から一部を修正・割愛させていただきました．

「プログラマ内輪話」ボード

■究極のドキュメントは？／匿名希望
某月某日

以前なにかの本で，「究極のドキュメントは，ソースリストだ！」って話を読んだせいか．現在制作中のシステムでは，コメントの書式から制定して構築に入りました．

ところが，こいつがたいへんなんだな……．かえって初心者には見づらくなり，ステップ数は膨れ上がって，コーディング時間が倍増し，ソース保管に圧縮ユーティリティを使用せざるをえなくなってしまった．

これで，できあがった実行コードも加わったら，とんでもない量になるな……．

かえって保守性を悪くしたみたいだ……．でも，私には，随分整理されたソース群になって，使いやすくはなっているのだけど……．うーん，コメントも付けようによっては，ただのゴミだな，こりゃ．

教訓：人の迷惑考えよう．

み － MYU － Umasoh　（かもしれない……）

■アウトラインプロセッサ型の設計支援ツール／匿名希望
某月某日

某マイコンメーカー（と言っても私が書くとわかる人はわかるかもしれない）に，アウトラインプロセッサ型の設計支援ツールがあります．文章を書く感覚でモジュール（章・節・項）を分けていって，ある程度細かくなったら，コーディングしたりライブラリを使ったりするのだそうです．

私は使っていませんので，細かいことはわかりません．はたして，使いものになるのでしょうか？（ならないだろうな）

Ａｉｃｅ（暫定）

■Knuthの文芸的プログラミング？／匿名希望
某月某日

ドキュメントをソースコードと一緒に記述しようって運動は，Knuthの文芸的プログラミングだっけ（？）とかが極端なレベルでやってましたね．

ソースとドキュメントを同じファイルに書いておいて，ドキュメントを抜き出して清書するプログラムや，ソースを抜き出すプログラムなんかを駆使していたような．

昔，PDP-11のOS，RT-11を使っていたときに，sysgenコマンドってバッチファイルだけど，があって，大部分がコメントというか解説文書になっていたのが印象に残っている．プリントアウトして見ると，何十枚にもなってしまうバッチというのも驚いたけど．

一緒に記述するなら，WEBの例に習って，関数仕様書部分とかは簡単に別に抜き出せるようにしておいたほうが良いかもしれない．grepとか使ってね．

■ギャグを混ぜながら……／匿名希望
某月某日

うちの会社にこの手の支援ツールというか，環境があります．ソース（COBOL）を日本語で書いて，コンパイルするときに辞書を見ながら通常のソースに変換して，本当のコンパイルするってヤツなんですけど．後，フローも書いてくれるらしいが……(なんて言うんだろ，普通は???)．実際は辞書作るだけで大変なので，誰も使ってません（笑）でもソースは見やすいと言えば見やすい．

コメントって書きすぎるとうるさいし，(結構見る気がなくなる（笑))書かなすぎるとなにやってんのかわからんし…….

私はギャグを混ぜながらいっぱい書くJACK'S OFFICE

「ドラゴンポスト［Free talk］」ボード

■翔泳社編集部御中／FV0119IDT
1992年2月11日火曜日午前0時11分

今日，「IBM DOS J5.0/V技術解説書」を買ってきました．まあ，これの読みやすいの何のって言ったら，市販本いらずという感じ．後，「BIOSインターフェイス技術解説書」もありますが，これも必須．

これらは，DOS5/Vを動かす上でのバイブルとなることでしょう．

これ迄，翔泳社刊「THE PROGRAMMER'S PC SOURCEBOOK」が頼みの綱だったんですが，いかんせん初出が古い．国内で出たのは1990年だけど，原本は1988年．元々，

米マイクロソフトプレスから出ていたものを，富士ソフトウェアとの協力で訳編したものなので，その時間差でしょうか．

今ではかなり知られている非公開ファンクションも載っていなかったりして，多少時代遅れの感があります（その割に，これまで非公開の筈の「DBCSの先行バイト数を求める」が堂々と掲載されていて，面白かったりする）．

BIOSにしても，DOS/V上で走らせる為には，最新の情報が不可欠．とは言え，丸っきりPC/ATの世界に無知だった私を導いてくれた，父の様にも思える本でありまして，ぜひとも父にはいつまでも健在であって欲しい処です．

どうでしょう，この際ですから翔泳社で独自につくっちゃいません？　新しい奴を．翻訳じゃなくて．

え？「ナツメ社が似たようなの出すからいいんじゃないの」？やだ，本の背に「SE」って入ってなきゃ，やだ（それに，あそこのは4.0の時に痛い目にあったし）．

うう〜ん，やっぱり駄目かな　FV01191DT　でした

■泣き言をいわせてね．／TACO

1992年2月11日火曜日午後11時3分

えと，会社で後輩に仕事のオイシイ部分をまかせてあげたら，本人は，厄介な部分を押しつけられた，と思ったらしくて，反発をくらってしまったんです．で，ちょとショックを感じて，俺も若い頃はそうだったかなあ，と反省モードで過去をふり返ったりしてたんですよ．

Taco

■ヒトデは海辺に一杯いるのに……／TACO

1992年2月14日金曜日午前0時18分

直接の原因は人手不足なんです．現在の部署は，小生とその後輩だけなので，日常の職場環境には，けっこう気を遣っています．

だからといって，若いのを甘やかしてるわけではないんです．ただ，早く一人前になって欲しくて，そのためには，仕事の面白い部分をどんどんやらせるのが早道だとも考えています．

ヤリガイのある仕事には，多少の責任はついてきます．で，こちらのオイシイと考えている部分は，彼にとってはその責任だけが重く映って，あんたが責任者なんだから，あんたがやりゃあいいんだ，とくるわけです．

自分の責任を部下に押しつけて済ませてしまう，そんな上司がデカイ顔をしている我が社の社風にも問題があるんで，一概に彼を責めるのも気が引けて，結果，歯切れが悪くなってしまうんです．反省．

でも，本当にショックだったのは，価値観の相違というか，誰もが楽しんで仕事をしているわけではないんだ，ということを，あらためて知らされたことだったんです．

小生はけっこう好きで今の仕事をしてるんですが，周囲の人間もきっとそうなんだ，というヘンな思い込みに，知らず知らずハマっていたようです．なんていうか，できるようになると仕事は楽しくなるという考えですね．そんな自分にガクゼンとしちゃった．〈^^;〉

かえって不自然に聞こえるかもしれないけど，収入のために好きでもない仕事を続けるというのが，なんかこうピンとこないんですよね．

長くなってゴメンネ．Taco

■仕事をする本質は変わらない筈／MYU

1992年2月14日金曜日午前7時11分

うーん，そのなんて言うか……．ただ，自らその仕事を選んでいるということは，自らにかかる責任も選択したことに成ると思っているんだけど……．

価値観の違いだけで済まされないとは思ってるんですよ……．新人にしろ，ベテランにしろ，仕事をする本質は変わらない筈ですけどね．もしそれがいやなら「さっさと仕事を変えんかい！」ってなもんです．“給料が高くなきゃだめ”とか“休みが多くなきゃだめ”とかね，贅沢を言うのはおかど違いで．あくまでも仕事をした結果への付加価値であると思っているんですけど．

ああ，朝っぱらなんで意見がまとまらない……．またちゃんとRESします．

【み】　- MYU -　（で，いいかな？）

■興味を持ってくれれば……／YODA

1992年2月14日金曜日午前8時4分

結構，そういう人，増えていると思いますよ．うちなんかは，中途採用者が中心なんですが，30代半ばの初心者さん（キーボードなんて触ったことが無い）なんかがくるんだけど，前のところよりは，給料がいいってだけなんですよ．で，超超丁寧に教えても，全然覚える気がなくって，超困りもの．まぁ，そういう人達は長続きしませんが．

さて，仕事分担だけどさ，一通り全部を説明して，興味を持ってくれたところをやってもらうようにしたらどうでしょうか？　私の場合は，いつもその手でやっていますが，興味を持ってくれるところが無いときは，やらせたいところに興味を持つように説明してみては？

Tacoさん，がんばってね．

YODA

PRODUCTS REVIEW

海老原浩之

Microsoft FORTRAN Professional Development System Version 5.1

マイクロソフトからProfessional Development System（以下PDS）シリーズに，MS-C Ver6.0，MS-BASIC Ver.7.1に続いて新しい仲間が加わった．それが今回紹介をする「Microsoft FORTRAN Version 5.1」（以下MS-FORTRAN Ver.5.1）である．このMS-FORTRAN Ver.5.1と同時期にMicrosoft Macro Assembler Version 6.0（以下MASM Ver.6.0）もPDSシリーズとして発売された．今回発売になったMS-FORTRAN Ver.5.1は，従来のVer.4.1に比べてかなり言語としての仕様が拡張がされており，MS-Windows Ver.3.0への対応，DECのVAX系やIBMの汎用系（VS）のFORTRAN言語と互換性がとれるようになった．では，MS-FORTRAN Ver.5.1を簡単に紹介していくことにする．

おもな特徴

MS-FORTRAN Ver.5.1の特徴として以下の内容があげられる．

- 言語仕様の拡張
- プログラミング環境としてPWBが搭載
- 複数言語によるプログラミング（MS-C Ver.6.0，BASIC Ver.7.1）のサポート
- グラフィックライブラリのサポート
- MS-Windows Ver.3.0への対応
- DEC VAX VMS, IBM SAA/VS互換
- MS OS/2への対応

またMS-FORTRAN Ver.5.1は，ANSI 77 FORTRAN，JIS FORTRANの上位水準に準拠し，FORTRAN90の拡張機能をサポートしたOptimizing Compiler言語である．では，上記の特徴について解説していこう．

言語仕様の拡張

MS-FORTRAN Ver.4.1と比べて言語仕様が大幅に拡張された．ここでは，Ver.4.1と比較して変更された項目，追加された機能について述べる．

動的配列が使える

コンパイルするときは，次元数だけを設定しておき，プログラムを実行するときに実際のサイズを決めてメモリから割り当てることが可能になった．

ファイルのアペンドが可能

ファイルのオープン時に，「ACCESS='APPEND'」と書くことにより，ファイルをアペンドモードでオ

ープンすることができる．

配列演算ができる

配列に対して，直接各種の演算を行うことができるようになった．ただし，各演算はスカラ演算で，配列間の演算はそれぞれが厳密な意味で同次元数，同要素でなくてはならない．

動的変数を扱える

C言語のように動的変数を扱えるようになった．FORTRANでは，通常は静的に変数が割り当てられる．しかし，Ver.5.1からは，AUTOMATIC文によってスタック上に割り当てることが可能になったのである．

条件コンパイルができる

$IF, $ELSE, $ELSEIF, $DEFINE, $UNDEFINEといったメタコマンドを使うことで，条件コンパイルをすることができるようになった．

.XOR.(排他的論理和)演算子

.XOR.が追加されたことで明確になった．機能は従来からあった．NEQV.と同じである．

IMPLICT NONE文

IMPLICT NONE文によって，暗黙のデータ型を宣言しないようにすることができる．仮に明確な宣言がされていない変数があった場合は，コンパイル時にワーニングメッセージを出す．

INCLUDE文

メタコマンドの$INCLUDEと同様で，指定したファイルの挿入をする．

$LOOPOPT，$NOLOOPOPT

ループの最適化をON/OFFする．

NAMELIST文

NAMELIST文により複数の変数にグループ名を与えることができる（リスト1）．それによって，READ文やWRITE文でグループ名を使った簡単な記述ができる．WRITE文では，各変数名とその値が出力され，READ文では各変数に値が代入される．

整数型への論理演算の対応

論理型だけでなく，整数型でも倫理演算子が使えるようになった．整数型の場合の論理演算では，各ビットに対して行われる．

制御構造文

新たに制御文としてSELECT CASE文が使えるようになり，ループを制御するための文もいくつか追加された．

SELECT CASE文はC言語などでおなじみの分岐のための制御文である（リスト2）．

CYCLEは，DOまたはDO WHILEなどで，制御を先頭に戻す．

DO WHILEは，条件が真の間繰り返す（リスト3）．

END DOは，DOまたはDO WHILEループの終りを示す．

EXITはループを終了する．

ユーザー定義型（構造体，共用体）

構造体と共用体が使えるようになった．構造体は，STRUCTURE文で構造体を宣言し，RECORD文を使って変数の定義を行う．共用体は，UNION文とMAP文を使って宣言する．

PWBの搭載

PDSシリーズには，プログラミング開発の統合環境としてProgrammer's WorkBench(以下PWB)が提供されている．このPWBによって，FORTRANを含むミックスドランゲージによる開発も行えるようになった．PWBについては，本誌1991年6月号掲載の記事「MS-C 6.0の統合開発環境」や専門の書籍が出ているので，それらを参考にしてほしい．

リスト1

```
INTEGER I1*1,I2*2,I4*4,IARRY(3)
REAL    R4*4,R8*8

NAMELIST /TEST/ I1,I2,I4,R4,R8,IARRY <---グループ名定義

I1 = 10
I2 = 20
I4 = 40
R4 = 100.0
R4 = 201.5
IARRY(1) = 11
IARRY(2) = 12
IARRY(3) = 13

WRITE(*,TEST) <---グループ名を使ったWRITE文

END
```


ワンラインNEWS
コクヨはパソコンを活用したシステム開発事業に乗り出す．米アップル・コンピュータと販売提携し，Macintoshを使った情報オフィスの企画提案をはじめる．

また，MS-FORTRAN Ver.5.1には，XMSメモリに対応した最新のCodeView3.14が付属している．

複数言語によるプログラム開発(MS-C Ver.6.0/MS-BASIC Ver.7.1)

MS-FORTRAN Ver.5.1では，MS-C Ver.6.0やMS-BASIC Ver.7.1，QuickBASIC Ver.4.5で書かれたプログラムを呼び出したり，逆にMS-C Ver.6.0やMS-BASIC Ver.7.1からMS-FORTRAN Ver.5.1のプログラムを呼び出すことも可能である．また，これら複数言語を利用したプログラム開発を，先ほど説明したPWB統合開発環境で行ったり，CodoViewを使ってソースレベルでのデバッグをすることも可能である．

グラフィックスライブラリ

MS-FORTRAN Ver.5.1では，MS-C Ver.6.0と同等のグラフィックスライブラリが提供されている．このグラフィックスライブラリを用いることで，DOS用(GRAPHICS.LIB)の場合は円や多角形などの図形表示，テキストウィンドウ，フォントを指定したテキスト出力が行える．またMS OS/2用(GRTEXTP.LIB)として，テキストモードで使うことができるルーチンが提供されている．

リスト2

```
     ERROR
          SELECT CASE(readerr)
                   .
                   .
                   .
                   CASE(writeerr)
                   .
                   .
                   .
                   CASE DEFAULT
          END SELECT
```

リスト3

```
     DO WHILE(count.GT.0)
         ....
         ....
         ....
         IF (COUNT.EQ.10)CYCLE
         ....
         ....
         ....
         IF (COUNT.EQ.30)EXIT
         ....
         ....
         ....
     END DO
```

MS-Windows Ver.3.0への対応

MS-FORTRAN Ver.5.1では，MS-Windowsへの対応がなされている．この機能は今回新しく追加されたQuickWinライブラリによって実現された．このQuickWinライブラリを使うことで，従来のFORTRANのソースに変更を加えなくてもWindowsに対応することが可能となる．QuickWin関数の追加により，Windowsアプリケーションらしいプログラムに変更できる．また，Windowsに対応することでDOS特有の640KBのメモリ制限にも悩まされずにすむようになった．

しかし，開発環境に関しては従来のDOS環境化で行うもので，VisualBasicやQuickC for Windowsのように完全にすべての開発をMS-Windows上で行う環境は提供されていない．ただ1つ，CodeView for Windows (CVW)だけがWindows用のツールとして付いているだけである．

また，QuickWinライブラリを用いて行うWindows対応とは，完全なWindowsアプリケーションを作成することではない．あくまでも標準の入出力をそのままウィンドウに表示するといったものである(写真1)．完全なWindowsアプリケーションを作成するためには，MS-C + Windows SDKを使用し，FORTRANで作成したプログラムをサブルーチンとして使用することになるだろう．

そのほかにMS-Windows Ver.3.0対応として，MS-FORTRAN Ver.5.1で作成されたプログラムを

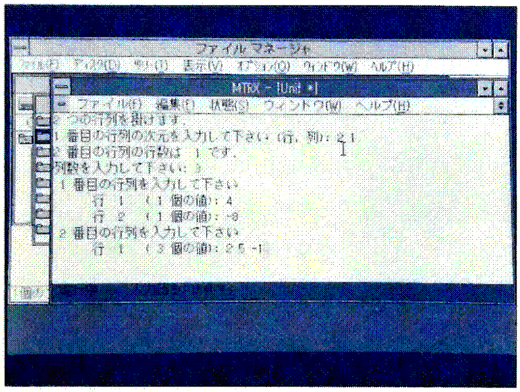


写真1
Windows対応ソフトも
作成できる

WindowsのDLL（ダイナミックリンクライブラリ）にすることも可能である．これらの機能はコンパイル時のオプションスイッチの指定で決めることができる（表1）．

DEC VAX VMS, IBM SAA/VS互換

MS-FORTRAN Ver.5.1では，コンパイル時のオプションスイッチの指定によってDECのVAX VMS FORTRANやIBM SAA/IBM VS FORTRANと互換性をとれるようになった．これにより，汎用機レベルで動作していたFORTRANのプログラムを簡単な修正をするだけで，パーソナルコンピュータ上で利用できるようになった．これらの互換を実現するオプションの設定を表2に示しておく．

表1

オプション	機能
/Aw	WindowsDLLのコンパイル
/Gw	WindowsDLLのエントリポイントのコンパイル
/MW	QuickWinアプリケーションのコンパイル

表2

オプション	機能
/4Yi	IBM SAA拡張機能を有効にする
/4Yv	DEC VAX拡張機能を有効にする
/4YV	IBM VS 拡張機能を有効にする

MS-OS/2への対応

MS-C Ver.6.0，MS-BASIC Ver.7.1などのPDSシリーズの言語と同様に，MS-FORTRAN Ver.5.1でもMS-OS/2に対応したプログラムの開発が行える．環境としては，MS-OS/2 Ver.1.2x以降が対象となっている．

さて，OS/2に対応することのメリットは，MS-DOSがもつ640Kのメモリ制限がなくなることだろう．これによって，大きな配列を定義することも可能であり，先ほど紹介したDECのVAX FORTRANやIBM SAA/VS FORTRANのプログラムを移植して動作させるにも好都合だといえる．

また，OS/2対応なのでマルチタスク，スレッド，OS/2用のダイナミックリンクライブラリの作成も行える．当然のことながら，それらをサポートするためのOS/2の標準システムルーチン（API）に対応するインクルードファイルやライブラリも提供されている．

そのほかに，DOSでもOS/2でも動作させることができるBUND実行型のプログラムも作成できる．

最後に

非常に簡単で足早にMS-FORTRAN Ver.5.1を紹介してきたが，今回のいちばんの要点としては，ANSI77 FORTRAN準拠またはJIS FORTRAN上位水準，MS-Windowsへの対応，MS OS/2へ対応，DEC VMS，IBM SAA/VS，FORTRAN90などに対応したことだろう．これによってダウンサイジングを意識したシステム構築が可能になった．

今までは汎用機でしかできなかったことでも，今後はパーソナルコンピュータである程度補うことが可能になったわけである．ダウンサイジング流行の昨今，MS-FORTRAN Ver.5.1は十分有効な武器になってくれるだろう．

動作環境

実行ファイルの作成に必要なシステム

・ハードウェア

80286以上のCPUを搭載したPC-9801シリーズもしくはPC-98シリー

ワンラインNEWS
沖電気工業は「MS-Windows 3.0」に対応したパソコンOCRソフト「読取名人」を発表した．Windowsを搭載した同社のパソコン「if」シリーズ上で利用できる．

ズ，PC-H98シリーズ(ただしハイレゾリューションモードを除く)

・OS

MS-DOS Ver.3.1以上または，MS OS/2 Ver.1.1A以上

・必要メモリ

640KB (MS-DOS)

空きメモリ1MB以上 (MS OS/2)

・ディスク容量

1MB以上のフロッピーディスク装置1台以上と最小3MB,最大11MB以上の空き容量があるハードディスク装置

・推奨システム

1MB以上のXMS Ver.2.0対応のXMSメモリ(MS-DOS)

1MB以上のLIM Ver.4.0対応のEMSメモリ(MS-DOS)

1MB以上のキャッシュディスク(MS-DOS)

80287/80387

マウス

・使用可能な日本語入力システム

MS-KANJI API使用の日本語入力システム(ATOK6/7の場合は，付属のドライバ(MSKN.COM)で対応可能)

作成された実行ファイルのターゲットシステム

・ハードウェア

80286以上のCPUを搭載したPC-9801シリーズもしくはPC-98シリーズ，PC-H98シリーズ(ただしグラフィックスライブラリを使ったプログラムは，PC-98LT，PC-98HAおよびすべての機種のハイレゾリューションモードを除く) *注1

・OS

MS-DOS Ver.2.11以上

MS OS/2 Ver.1.1A以上

MS-Windows Ver.3.0以上(リアルモードを除く)

・必要メモリ

システムと作成されたプログラムに依存

注1：QuickWinを使用したアプリケーションの場合は，Windows環境に準ずる．

パッケージ内容

・ディスク

7枚

・マニュアル

Install Guide

Programming Guide

Reference

Environment and Utilities

Quick Reference

・ユーザー登録カードなど

■Microsoft FORTRAN Version 5.1 Professional Development System

価格：98,000円／発売元：マイクロソフト株式会社／電話：03-3363-1201

Book Review

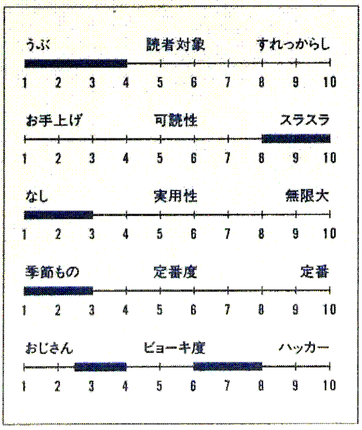
■中村正三郎

薀蓄のネタ本としても面白いコンピュータ界のジョーク集

■笑うコンピューター

Karla Jennings著
邦つゆこ訳
1500円
技術評論社

■評価チャート



コンピュータ界のジョーク集である．微笑ましいエピソード集である．ぞっとする怪談集である．

著者Karla Jenningsは，コンピュータ界のさまざまなジョークを，時には歴史の闇から掘り起こして，精力的にかき集めた．それらを本書では10のジャンルに分類して編集している．

個々のエピソードは短く，テンポよく読み進めることができる．ジョーク集を訳すのは大変な作業であるが，本書はかなりの程度成功していると思う．

初めは，ハッカーに関するジョークを集めた「ハッカーワールド」．コンピュータにのめり込んだ人たちの楽しさ，滑稽さ，悲しさがスケッチされる．ここに出てくるMITのスッスマンとは，先月の最後に紹介した本の著者であり，Schemeで有名なサスマン教授だと思うのだが，訳者の間違いだろうか．

「コンピュスピーク」の章は，コンピュータ界ならではの言葉遣いを扱っている．ガベージコレクションの研究資料を，ガベージ(ごみ)と書いた箱に入れていたら，用務員に捨てられそうになった話など，さもありなんと思える，一般人との語感のずれを軸にした章だ．UNIXという言葉は非常に意味深な言葉であるとして，宦官と間違われた例をあげている．評者は昔，ホモセクシャルを強く連想させる言葉であると聞いたことがある．

病的な言葉遊びの集積として「ハッカーズディクショナリー」を紹介している．これは新版がedited by Eric Raymond『The New Hacker's Dictionary』(The MIT Press)として出ている．

この章には，Pascalの設計者ヴィルト教授が自己紹介のときに使うジョークが出てくる．

ヨーロッパでは，ニクラウス・ヴィルトと名前で呼ばれるが，アメリカではニックルス・ワースといわば値で呼ばれ，その値はたったのニックル(5セント)の価値しかないと笑わせるものだが，この訳注はおかしい．

訳注は，発音の違いをネタに，さらに変数の名前と型を宣言してから，値を代入するPascal言語のプログラミング作法にも引っかけていると書いているが，誤った解釈だ．

名前呼び(call by name)と値呼び(call by value)は，プログラミング言語一般に見られる引数渡しの機構であって，これに引っかけているのだ．

「追伸：ねずみは灰色」は，ウィーナー，ノイマンといった大御所にまつわるエピソード．比較的知られているものだった．

「悪魔か神か」はSFが昔から扱ってきたコンピュータテーマの話題である．中でELIZAをエリザと訳しているが，一般的にはイライザではないか．

「会社と寝室で」は，オフィスと家庭でコンピュータに引っかき回される話である．この章から以降，ほんわかユーモアの微笑ましい話は少なくなり，だんだんとエピソードの病的な度合いが進む．章の題名をあげれば，「吸血ウイルス」「諜報と情報」「カネ，カネ，カネ」「モンスターの逆襲と屈服」となる．

題名からおわかりのように，犯罪にまつわるエピソードが増えるからであ

ワンラインNEWS
ジャストシステムは，ロータス社の表計算ソフト「1-2-3」で作成した表やグラフを一太郎Ver.4で取り込む機能拡張プログラム「1-2-3ファイルコンバータ」を発売した．

る．コンピュータが原因となって人が死ぬエピソードが登場するからである．

最後の章，「コンピュータ革命の歴史」は，バベッジとエイダ・バイロンのころから始まり，ENIACなど現代のコンピュータの原型ができるまでの歴史を概観している．本章の白眉は，世界初の電子式デジタル計算機の発明者ともいわれる，栄光なき天才アタナソフの悲劇を追った箇所である．IBMが1950年代の初め，コンピュータ市場をどう考えていたという話は非常に興味深かった．

コンピュータ界のジョークを多く収録，整理した手軽な読物としても，また蘊蓄のネタを仕込む本としても，本書は面白い．

続々登場する春の新刊総めくり

古田島義和（神田・書泉グランデ）

今回は，新刊をできる限り載せたいので，余計なことは差し控えます．はじめに，前々回，Microsoft Press社のWindows 3.1のマニュアル本を紹介しましたが，それ以外にたくさん出版が予定されているので紹介します．()内は著者名です．

Howard Sams社「Learning Windows 3.1」(D. Derrick)，「Tricks of the Windows 3.1 Masters」(E. Tiley)，「Windows 3.1 Bible」(B. Sosinksy & R. Mullin)，「The Best Book of Windows 3.1」(C. Townsend)，「The First Books of Windows 3.1 2/E」(J. Nimersheim)，Brady社「Fully powered Windows 3.1 2/E」(B. Alperson)，Que社「Windows 3.1 Programmer's Reference」(J. Mccord)，「Using Windows 3.1」(R. Person & K. Rose)，「Windows 3.1 QuickStart」(同)，「Windows 3.1 Quick Reference (Que Development Group)，「Programming in Windows 3.1」(T. Farrell & R. Comally)，New Riders Publishing社「Inside Windows 3.1」，「Windows 3.1 On Command」，「Windows 3.1 Networking」以上の4社は，最近Prentice Hall (以下PH) の傘下に参入，パワーアップをはかっています．ほかにAddison Wesley (以下AW) から「Windows 3.1 At Work」(M. Campbell) と，ユーザーにとってはうれしいニュースです．

BORLAND C++がバージョン3.0になり，Howard Sams社「Mastering BORLAND C++」(T. Swan)，Que社「Using BORLAND C++ 3 2/E」(L. Atkinson & M. Atkinson) が出版，そのほかにQue社のBy Example Seriesの中にも「BORLAND C++ By Example」(J. Purdum) があります．

このシリーズはあと4点あり，「C By Example」(同)，「Turbo Pascal 6 By Example」(J. Jesus)，「Visual Basic By Example」(D. F. Scott)，「QBasic By Example」(G. Perry) と，各ユーザーにとっては必読書となりそうです．

つぎにOSの新刊を数点ご紹介しましょう．Berkeley UNIXの新しい環境を知ることのできる図書として，PHから「The Berkeley UNIX Environment 2/E」(N. Horspool)が出版されました．本書には，ANSI Cによるプログラミング実例集やいくつかの新しい章が加えられています．SUN OS 4.1 Operating System，4.3BSD，ULTRIXなど，さまざまな環境を網羅した解説書で，UNIXユーザーに最適な1冊でしょう．

日本の読者には周知のA. Tanenbaum氏の「MINIX 1.5：for the SPARC」が，Prentice Hallから「MINIX1.5」シリーズの新刊として出版されました．本書には，8枚のディスクにソースコードが収められ，Free Software FoundationのGNU C Compilerを使用した注目のソフトパッケージとなっています．有名といえば，TCP/IPでおなじみのD. Comer氏とO. Stevens氏のコンビが，PHから「Internetworking Software with TCP/IP：Vol II－Design, Implementation and Internals」を発表，近々入荷の予定です．

最後に，X-Window SystemがVer.11 Release 5にリリースが変わり，O'Reillyの解説マニュアルシリーズや，ほかの出版社から次々に発表，入荷されますので，これについては次回でご紹介できればと思います．

■和書&洋書ベスト10

（書泉グランデ調べ：1992年1月1日～1月31日）

	和書	出版社
1	スーパーエンジニアへの道	共立出版
2	Cプログラマのためのアルゴリズムとデータ構造	ソフトバンク
3	追補版SX-WINDOWプログラミング	ソフトバンク
4	UNIXオブザーバー92.2 No.3	スペック
5	sed&awkプログラミング	アスキー
6	電子立国　日本の自叙伝（中）	日本放送出版協会
7	ResEdit完全版	トッパン
8	98シリーズで使うVisual Basic入門	ビレッジセンター
9	WTERMで今日から始めるパソコン通信 改訂版	秀和システム
10	NIFTY-Serve '91年度版	星雲社

	洋書	出版社
1	Learnig GNU Emacs	O'Reily
2	Programming the Display PostScript System with NeXT Step	Addison Wesley
3	Standard C Library	Prentice Hall
4	Object-Oriented Modeling and Design	Prentice Hall
5	Eiffel:The Language	Prentice Hall
6	GNU Emacs	Addison Wesley
7	UNIX Network Programming	Prentice Hall
8	Compilers (paper)	Addison Wesley
9	MS,MS-DOS Prograwmmer's Reference	Microsoft Press
10	Art of Compiler Design	Prentice Hall

★　★　★ 佐々木哲の ★　★　★

U.S. プロダクツウォッチ

★軽快さに期待，Pen/GEOS

予想された動きではありますが，よりよいWindowsというべきGeoWorksで評価の高いGeoWorks社*1はこの軽快なGUIのPen対応の開発を発表しました．Pen Windowsなどとは違いサイズも小さく，また8088，8086CPUでも動作します．Pen/GEOSという名前で今年後半に500ドル以下で提供されるということです．この発表には8086ベースの1チップPCを発売しているChips&Technologiesも名前を連ねていました．

このPen/GEOSにより今までのGeoWorksアプリケーションがPen環境で使えるだけでなく，すでにPen/GEOS専用のアプリケーションの開発も始まっているということです．

Pen/GEOS自体はすでに英語だけではなく，フランス，ドイツ，イタリア，オランダ，スペイン，フィンランド各国語のバージョンが40万コピー以上販売されたそうです．

*1：GeoWorks, Inc.
2150 Shattuck Ave., Berkeley, CA 94704
phone: (415) 644-0883, fax: (415) 644-0928

★フォントの種類が多すぎる

日本でもWindows用などにさまざまなアウトラインフォントパッケージが手に入るようになってきました．値段が高い，フォントの品質がまだまだばらばら，などいろいろ問題も見られます．今数社からフォントラスタライザという形でドライバが出て，それぞれ用のオプションのフォントがまた多種多様に出ています．

これらがまたそれぞれフォントのフォーマットが異なるのですね．軽いラスタライザ，美しいものと使い分けたくても（もっとも値段も高くてそこまで余裕のある人も少ないでしょうが），それぞれにフォントフォーマットが異なり使い回しができない．こちらのラスタライザ用に欲しいフォントが出ていてもこちらのものでは使えない．以前このコラムでも触れたように，グラフィックファイルフォーマットのようにフォントのフォーマットの互換性のなさが混乱を生み始めました．

USではどんな状況でしょう．やはりフォントに関してはAdobe，Bitstreamと各社各様のフォーマットが存在します．Adobeのもののようにフォントのフォーマット情報が完全には公開されていないものもあります（これもType 1フォントの情報が公開されてだいぶよくなりましたが）．それでもお互いのフォントを変換して使えるような環境が揃ってきました．ここではその中でも新顔のものを紹介しましょう．

Atech SoftwareのAllType*2はMS-Windows，DOS上のいかなるフォントでも変換し，使えるようにするフォントのコンバータソフトウェアです．Adobe PostscriptのType 1とType 3，MS-Windows 3.1で使われるTrueType，Agfa CompugraphicのIntellifont，BitstreamのFontware，最近伸びているDigital Typeface CorporationのNimbus Q，そしてAtech Software FastFont．これらのフォーマット間で双方向で変換可能です．フォントを扱う際に品質に影響を与えるヒントもきちんと変換されます．さらにフォントを変換する際にボールドや縮小，拡大，変形などさまざまな処理を行い，ウェイトやピッチ，イタリックなどの変形処理を行うことができます．

MS-Windowsだけでなく，DOSベースのソフトでもさまざまなフォントが使われているUSですが，このコンバータではそれぞれのソフトにあったフォントフォーマットが容易に自動的に選択されるそうです．

*2：AllType for DOS
AllType for Windows
どちらも$79.95
Atech Software
phone (619) 438-6883

★便利な電話サービスとは?

日本の電話の近年の大きな話題はISDNでしょう．NTTは戦略的な低価格でサービスを提供しつつあります．それでも交換機の整備など息切れしてきているようですが．

USではRBOCと呼ばれる各地域の電話会社が分割されており，さらにT-1回線など高速な通信線サービス，各種の付加価値サービスが以前から充実していたせいもありISDNはそれほど話題にもならず，普及もしてい

ワンラインNEWS
アップルコンピュータは，同社製品のサポート・コンサルト業務を行う「アップルセンター」を広島市など4か所に新設した．

ません．日本のように個人の家にISDNなどしばらくは想像もつかないような状況です．各種日本では見られないサービスのうちCaller IDを見てみましょう．

これは電話会社と別に契約することにより，一般の電話回線で電話を掛けてきた相手の電話番号を知ることができるサービスです．Caller IDアダプタを購入し電話機と電話線の間に取り付けます．Caller IDを契約している場合，電話が掛かってきたときに最初に1200 bpsの半二重による独自の手順により電話番号が伝えられ，Caller IDのアダプタに表示されます．電話会社の売り文句としてはいやがらせの電話などを防ぐことができ，プライバシーの保護にあたるということですが，やっぱり使い道はセールスのツールです．たとえば，大きな規模で使うなら，何かの商品のプレゼントなどをTVなどで呼び掛け，いっせいに申し込みの電話を視聴者にしてもらいます．企画した側は電話を受けるだけでその商品に興味を持った人たちの電話番号のリストが人手も掛からずでき上がっていきます．いったん電話番号がわかれば後は名前，住所，各種個人データのデータベースと組み合わせるだけで全部わかってしまいます．この強大な力を持ったCaller IDサービスはサービス地域が全米に広がりつつあります．

このCaller IDサービスをPCで受けられるアダプタが発売されています．Zeus Phonestuffという新興の会社はWhozz Calling?[*3]（誰が電話してきているの？）という名前でPCのシリアルポートにつながるモデムのようなアダプタを発表しました．これをPC上のアプリケーションと組み合わせることにより，いろいろなことが可能になります．たとえばコンサルタント業務では掛かってきた電話番号により，各顧客の電話でのサポートの管理が簡単にできますし，登録されていない番号からの電話は受け取らないですみます．PC上のデータベースと組み合わせると電話が掛かってきた途端顧客の電話番号にしたがって顧客データを画面表示させることにより迅速で的確な対応ができます．電話とコンピュータの一味違った，強大な力を与える可能性のあるデバイスですね．

＊3：Whozz Calling? $79
Zeus Phonstuff
phone: (404) 587-1541, fax: (404) 587-1609

★ 6つの機能を1枚のカードに

IBM PC互換機が出はじめたころから，さまざまな拡張カードが出てきています．最低限でもパラレル，シリアル，ディスプレイ，FD/HDコントローラをカードで付けないと機能しないPC互換機．いくつかの機能を兼ねたカードがあります．ASTなどはSix Packという名前でパラレル，シリアルなどが付いたEMSカードを出して大成功しています．

AMT Computers[*4]は，同じく6つの機能を1つにしたカードを発表しました．ATバス用のSigma Six boardには，10MbpsのEthernet，9600/2400bpsのFAX modem，1024×768の解像度を持つSVGA，Flash ROMに入ったDOS，IDE HDコントローラ，FDコントローラ，そしてパラレル，シリアルポートが1つのカードになっています．これのEthernetの入っていないものもPersonal Editionとして用意されています．CPUとメモリが載ったマザーボードにこれを載せるだけでフルにオプションの揃った互換機のでき上がりです．DOSがFlash ROMに載っていることからもディスクなしの組み込みシステムにも最適でしょう．

＊4：Advanced Micro Technology Computers, Inc.
123 University Pkwy., Pomona, CA 91768
phone: (800) 487-4539, fax: (714) 598-6120

Illustration：Masami Nozue

Illustration : NOBUKI FUKUHARA

西川和久の

希望的観測

日本語Windows＋MMEその後……
日本語Windowsと海外製品のコンビネーション

1 Super Video Windowsがやってきた!!
残念ながら640×480 256色だけど

前回少しビデオ入力をWindows上に表示することができる〝Super Video Windows〟について触れたが，その後すぐに商品が届いてしまったのだ．ワクワクしながらセットアップしたのだが，残念ながら当てが外れてしまった．当初の目論見では1280×1024 256色の日本語Windows上でテレビ画面の窓が表示できるものと思っていたが，いろいろ接続して調べたところ「640×480の解像度でしか動かない」のだ．まさかSuper Video Windowsのために通常使うマシンの解像度を下げるわけにもいかずに，仕方がないのでメインのマシンへ組み込むことはあきらめ，サブのPS/55Z 5510Sへ入れることになった．ちょうどS3のチップを搭載したOrchid F1280が1枚余っていたのでこれを640×480 256色で使い，さらにIBM純正の16インチモニタへ接続した．あまり画面サイズが小さいと画像がよく見えないからだ．

オプションのテレビチューナーボードも購入したためWindows上のパネルからテレビのチャンネルまで操作することができる．これはこれでなかなか面白い．考えてみれば今まで社内でパソコン用のモニタは数多くあるが，テレビやビデオが見れるディスプレイは皆無であった．つまり見たい番組と残業が重なってもあまり気にしなくても済むようになってしまったのだ……(^^;

5510Sはi80386SX/16MHzのマシンであるために，日本語Windowsを動かすにはちょっとCPUパワー不足であるが，S3チップのおかげでそこそこストレスなくアプリケーションが使える環境になっている．

気を取り直して，高解像度でビデオ入力ができるカードを探してみたが比較的安価なものはすべてVGAのレゾリューションであり，海外広しといえども私が希望するようなカードはないようだ……残念．

2 HP LaserJet 3Pと日本語Windows
激安の海外プリンタで日本語処理

海外のプリンタで日本語を簡単に印刷する方法がある．あまり知られていないようだが，米国版Windowsに対応するプリンタドライバを日本語Windowsへ組み込み，本来漢字フォントを持たないプリンタでもWIFEフォントを含む漢字が印刷できるのだ．通常日本語Windowsのプリンタドライバはプリンタ側のフォントは使わず，すべてビットマップに展開して印刷しているため漢字フォントなどどうでもよい．唯一の例外はスケーラブルフォントを持ったプリンタで，かつ，ドライバでも対応している場合だけである．漢字Postscriptプリンタがその例であろう．

¥WINDOWS¥SYSTEMにDISPATCH.BINというファイルがあるが，このファイルはPrinter Dispatch Driverと呼ばれるもので，日本語Windows 2.1の時代

ワンラインNEWS
ソードはパソコン販売事業を拡充，LANシステムの販売などを担当するSEの増員，周辺機器の開発を通じ，売り上げ全体に占めるシステム販売の比率を引き上げる．

に作ったWIFE未対応プリンタドライバを簡単にWIFEに対応する目的で作られたのだ．ところが海外のプリンタドライバにもWIFE未対応という意味で条件的に当てはまるため，結果漢字が印刷できてしまう．

現在，米国ではレーザプリンタが非常に安くなっており，安い物だと実売で500ドル程度からある．国内で最も安いEPSONのLP-2000が198,000円だと考えるとその差歴然だ．さらに高級な部類に属すHP LaserJet 3Pでさえ1,500ドルしない．海外のレーザプリンタを国内で使った場合〝トナー〟の心配があるが，これはHPに限っては何も心配ない．実はエンジンはCanon製でありLBPシリーズのものがそのまま流用できるのだ．DOSアプリケーションからの印刷さえ諦めてしまえば，激安の海外プリンタでも日本語処理ができるというわけだ!!

HP LaserJetシリーズは純正/サードパーティー問わず，さまざまなオプションキットが用意されている．中でもすごいのが先月隣の佐々木氏が紹介していたが解像度をアップするUpgrade Kitである．標準では普通の300dpiであるが，このキットによって600dpiや800dpiにすぐさま変身してしまう．それもI/Oは国内のパソコン市場で高級すぎて全く無縁の〝ビデオインターフェース〟になってである．解像度が上がればそれに伴いデータ転送量が増大するがプリンタエンジン直結型のビデオインターフェースなら快適である．まだ実験していないので，このキットを搭載して漢字が印刷できるかどうかは定かではないが，先の理屈からうまく行けば本体と合わせても2,500ドルとしないプリンタで800dpiの世界に突入できる．恐ろしい世界である．

この話は日本語Windows 3.0なら，どの機種でも同じ理屈が成り立つため興味がある人は各自のリスクで実験して欲しい．ただし，何かあっても私やメーカーが保証するわけではないので，あしからずご了承ください…… (^^;

3 GUI環境の必需品…… 日本でも大型ディスプレイの時代に

高解像度日本語Windowsによって，日本でもやっと16インチ以上のディスプレイや，マルチシンクモニタの広告が多くなってきた．これらのモニタもほとんどがすでに海外で売られているものだ．パソコン本体は何度も買い直すが，モニタはなかなか変えないので，これを期に良いモニタを購入するといいであろう．これから先のGUI環境には絶対必要なアイテムである．パソコンばかりいくら速くても，モニタが悪いとその効果は半減する．

結局気がついてみると，日本語処理を行うべき日本語Windowsはなんのことはない，ほとんど海外製品を使った方が〝処理速度〟〝コストパフォーマンス〟〝解像度〟すべてにわたって圧倒的に有利になってしまうのだ．DOS/Vが発表された当初「黒船がやってきた!!」などとよくいわれたが，裏を返せば日本語Windowsこそそのものではないだろうか!?　たぶん今後出てくるさまざまな環境でも同じことがいえるであろう．〝漢字ROM縛り〟さえなくなってしまえば，平和な世界がやってくるのである．

Computer Laws

コンピュータの法律

懸賞募集したプログラムの著作権はどこにあるのか

◆

熊谷信太郎（弁護士）

■募集要項が基本になる

今回は，他人にプログラム作成を委託する典型的な場合とはやや異なりますが，コンピュータプログラムを懸賞募集した場合の著作権を考えたいと思います．

プログラムを懸賞募集した場合に，当選作に対して主催者が，賞金を出すことがありますが，この場合には，プログラムの著作権は主催者と，応募者のどちらに帰属するのでしょうか．それは，賞金の有無によって結論が異なるのでしょうか．応募作品は返還しないという条件がつけられている場合がありますが，その条件はプログラム著作権の帰属に影響があるのでしょうか．

プログラムの懸賞募集の場合に，プログラム著作権が誰に帰属するかは，基本的には募集要項に書かれた契約の内容によって決まります．

すなわち，募集要項の中に，当選作については，著作権を主催者が取得すると記載してある場合には，応募者はその条件を受け入れて応募したわけですから，プログラム著作権は主催者のところに移ります．

もっともこの場合，著作権が主催者のところに移っても，著作人格権は著作者のところに残るということを忘れてはいけません．

著作人格権とは，公表権，氏名表示権，同一性保持権のことを指し，これは，もともとのプログラム作成者のところに帰属しています．プログラム著作権を譲渡しても，著作者人格権までも移転するわけではないのです．

ただし，未公表著作物を譲渡した場合には，公表について同意したものと推定されますので，実際に問題となるのは，氏名表示権と同一性保持権です．

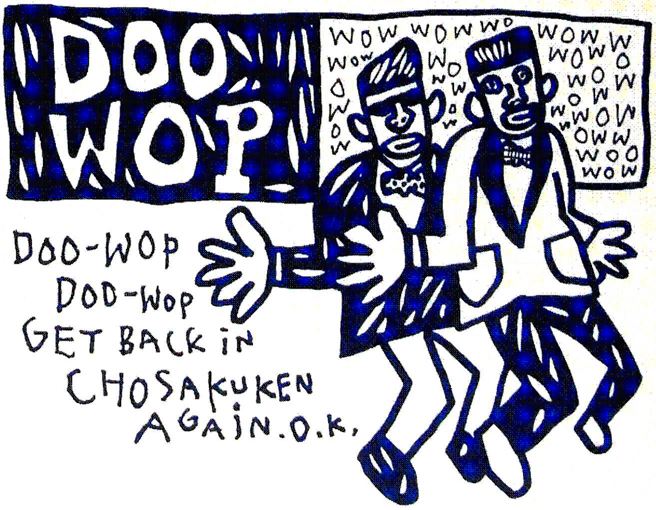
したがって，主催者がプログラム著作権を取得するという契約になっていたとしても，元の著作者に無断でそのプログラムに手を加えたり，著作者名を変えたりすることはできません．

■賞金が著作権に影響を与える

これに対し，募集要項でプログラム著作権の帰属について明示していない場合には，基本的に実際の著作者がプログラム著作権を持つことになります．

この場合，応募作品は返却しないという記載があっても著作権の帰属に影響はありません．それは，プログラムを記憶させている媒体であるフロッピーを返還しないというだけのことであり，懸賞小説の場合に，原稿を返還しないというのと同じ意味ですので，著作権の帰属には影響を与えることはありません．

これに対し，賞金の有無，金額の多寡はプログラム著作権の帰属に影響を与える要因になると言えるでしょう．賞金の額が多額であり，プログラム著作権の譲渡代金と評価できる場合には，プログラム著作権はプログラム製作者から主催者に移転することになります．賞金の額がそこまでいかない場



Illstration : Nobuki Fukuhara

ワンラインNEWS
東芝は大幅に小型・軽量化したワープロ「ルポJW01」を発売する．従来機に比べ大きさ・重さともに約半分とした．おもな販売ターゲットは大学生．

合には，それは単なる報奨金であり，プログラム著作権の帰属には影響を与えません．

どのくらいの金額までいけば譲渡代金と評価できるかは，プログラムの価値との関係で決まりますので，一概には言えません．原稿やカット，写真のネガ等については，買取りということがよく行われますが，それらの場合には，買い取り料は，基本的には，原稿やネガ等の所有権を移転し，それらの使用を許諾することに対する対価であると言われており，著作権譲渡の対価であるとは考えられていません．

それとの対比で考えても，プログラムの懸賞賞金が，いわゆる著作物の買取り代金程度の場合には，プログラム著作権の譲渡があったわけではなく，単にプログラムの使用の許諾があったにすぎないものと見るべきでしょう．

したがって，その場合には，主催者としては，プログラムの募集の趣旨に従って，当選作品を利用することはできますが，それを商品化するなどの，別途の利用をする場合には，プログラム著作者の同意を得る必要があるわけです．

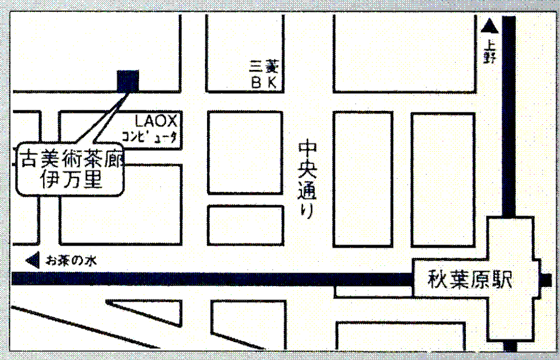
また，入選しなかった応募作は，仮にフロッピーを返還しないという特約があったとしても，プログラム著作権は応募者にありますので，もし，主催者がこれを勝手に利用した場合には，著作権侵害となり，応募者は差止めを請求することができます．

アキバ短信

古美術茶廊　伊万里

秋葉原の隠れ家

営業時間　7:30〜19:00
TEL　03-3251-8369



秋葉原は移り変わりの激しい街である．新しい店は次から次へとオープンし，古い店もしばらく見ない間に改装してすっかり様変わりしていることもしばしばだ．3年ほど前に翔泳社が刊行した「パソコン版標準秋葉原ガイドブック」という本を見てみると，3年間の変化はまことに著しいものがある．ラオックスコンピュータ館や，ツクモパソコン本店，ステップの2号店などが開店したのも，ラジオ会館の3階にもずらりとパソコンが並ぶようになったのも，万世本店が改装になったのも，神田市場が移転して駐車場になってしまったのも，すべてここ3年以内のことなのだ．

だが，3年はおろか古い伝統を引き継ぐものが多いのもまた秋葉原という街の特色でもある．この欄でも紹介した交通博物館は，かつては東京で4番目の乗降客数を誇ったという万世橋駅の跡地であるし，大手の電気店のひとつ，ヤマギワは昭和の初期には，すでに現在の本店がある場所で開業していたという．また，創業300年をほこる薬局なんていうのもある．今回紹介する「伊万里」も，そんな古くからの秋葉原を感じさせてくれる場所だ．

店内に一歩足を踏み入れると，まずは右側のケースにずらりと並んだ陶磁器類が目につく．ウェッジウッドでもロイヤルコペンハーゲンでもなく，古伊万里である．テーブルや椅子は木製の民芸調．腰をおろすと，おばちゃんがせんすに書かれたメニューをもってきてくれる．コーヒーはホットとアイスだけ，後は，紅茶，日本茶(お菓子つき)，ジュースなど．でも，いまどきどれも330円という値段は泣かせてくれる．「裏もありますからね」という声で，せんすを翻すと，反対側はみつ豆，あんみつなど甘党メニューである．カウンターの奥でコーヒーをいれてくれるのは，焼き物に負けない年期の入った(?)お婆さん．その横ではおばちゃんたちが世間話をしながら，白玉を丸めている．年期の入った伊万里焼のカップでコーヒーを飲めば，柱時計がボーンと時を打つ．

そんなのんびりした空間が，秋葉随一のパソコンのデパート，ラオックスコンピュータ館のすぐ目の前に存在する．伊万里の間口は狭いので，うっかりすると通り過ぎてしまいそうだが，奥行きは広いし，地下にも座席があるので，少人数ならグループでも大丈夫だろう．伊万里から神田明神下交差点に向かって歩くと，ラオックス・コンピュータ館の店頭展示品を安く販売するコンピュータ館II，さらに先には宮内庁御用達の箸(!!)を扱う箸勝本店もある．まさに新しいものと古いものが混在しているのである．

●プログラマの仕事場 13

株式会社エーエスピー

エディタFinalで知られたソフトハウスがGUIとマルチメディアの世界をめざす

株式会社エーエスピーという名前は知らなくても，エディタFinalといえば知っている人は多いはずだ．Finalは，さまざまなマシンに対応しており，パソコン16機種に対応したSuperPack，UNIX対応のX-Finalなどがある．同社をたずねて，アプリケーション開発に対する考え方や，最新のWindows3.1，WindowsのMME(Multimedia Extensions)などの動向について，小塚雅弘，小田島孝久の両氏からお話をうかがった．

UNIXへのこだわりからマシン2台でスタート

エーエスピーの設立は，いまから8年前のこと．最初はUNIXマシンが1台とPC-100が1台であったという．小塚氏は会社設立当初を次のように語ってくれた．「UNIXに興味があったので，System Vなどは早い時期から手掛けていました．当時，ある会社から小さいUNIXマシンを出したいという話がありまして，そのシステムコンサルティングのような仕事もしていましたね」．

「うちの会社では受託開発は一切やっていません．当初はクロス開発というのはUNIXで結構いけるのではないかと思っていたのですが，パソコンの場合は意外にセルフで開発できてしまう．UNIXが共通のプラットフォームにならずにパソコンが伸びてきた．それじゃあパソコンでアプリケーションを作ろう，パソコンならツールがいちばんいいだろう，ということになって，エディタを開発することになったのです．実はFinalの前にもREDというエディタがありまして，その前にはOS/9でもエディタを出していました．エディタについては全部に目を通すことのできるソースファイルを持っていたのです」．

Finalというと，パソコンユーザーにはパソコン専用のイメージが強いのだが，意外にもエーエスピーの最初のエディタはUNIX上で開発されたものだという．

「最初のエディタを作ったのはもう6年ほど前のことになります．Finalだけが特別だというわけではなかったのですが，最初はものすごくノロいエディタでした．それがパソコンのプラットフォーム用に書き換えることによって――いわゆるテキストVRAMに直接書き込むという方法をとったわけですが――自分たちでも使えるスピードになったわけです」．

「UNIXをクロス開発で利用することを考えていたのですが，結局Finalはパソコン上のセルフでの開発に使われることになりました．やはり，シリアル端末をキャラクタベースで使っているのでは，UNIXの普及は期待できません．ビットマップの表示できるワークステーションタイプのマシンが出

ワンラインNEWS
スリーコムはネットワーク構築用のLAN接続機器にアプリケーションソフトを組み合せたパッケージ営業を展開．商品知識が乏しいユーザーが導入しやすいよう配慮．

てきて，ようやくいけるなと思うようになりました」．

C言語への思い入れがUNIXに結びついた

「代表取締役の鶴澤と私は，以前はライフボートという会社におりました．当時ライフボートではUNIXはやらないという方針だったのです．UNIXにははじめからツールが組み込まれているので商売にならないだろうという見方をしていたのですね．そのころライフボートではBDS Cというコンパイラも手掛けていたのですが，ちょうどFORTHという言語もできたところで，Cをやったほうがいいのか，FORTHのほうがいいのか，会社も相当悩んでいたようでした．でも，アメリカでの評判を聞くとCのほうがよさそうで，WhitesmithsのCなどもでてきていました．私自身もCにほれこんでしまったのです．Cが好きならやっぱりUNIXだろうということでUNIXをはじめたわけです．ふつうとは逆の順序だったかもしれませんね」．

現在では，X-FinalをはじめUNIXのソフトウェアの売り上げが全体の65%を占めるまでにいたっている．

マルチメディア時代を迎えて新しいWindowsに期待

「元々はみな音楽をやっていたのです．最近になって，Windowsを一生懸命やっているのは，マルチメディア・エクステンション（MME）が出たからです．いまはまだMacにはかなわないですが，だんだんよくなっていくでしょうね．新しいWindows3.1には最初からMMEの要素がかなり取り込まれています」そういって，小塚氏はマルチメディア対応パソコン（MPC）上でWindowsを起動し，BGMを聞かせてくれた．

MMEとは，マルチメディア関係のAPIをWindowsに付加するための一連のソフトウェアである．アメリカでは，CD-ROMに納められたWindows MMEのシステムとして供給されている．今年出荷されるというWindows3.1には，マルチメディア機能がかなり取り込まれる予定だとも聞く．「マルチメディア」といってもピンと来ない向きには，パソコンで動画や静止画を表示したり，音声を制御するためのものと考えてもらえばよい．

パソコンをマルチメディア化するためのハードウェア仕様であるMPCレ

About Company

株式会社エーエスピー．代表取締役鶴沢幹夫．スクリーンエディタX-FinalをはじめとするUNIX上での各種ツールの開発・販売をおもな業務とする．社員数15名．

Location

京浜急行で品川から7つめ，競艇場があることで知られる平和島で下車，徒歩10分．地元の商店や住宅の立ち並ぶ一角にある．

Products

「X-Final」（98,000円）
NEWS，LUNA，EWS4800，HP9000，SUN3/4，AS3000シリーズ，S3，S4，NeXTなど各種UNIXワークステーション対応のスクリーン・エディタ．EUC/S-JIS/JISの3種類の漢字コードに対応．テキスト取り込み時に漢字コードを自動判別するだけでなく，漢字コードの変換やメニュー・メッセージなどの表示漢字コード変更などの機能を持つ．

「Final Super Pack」（28,000円）
パソコン用日本語スクリーンエディタ．ひとつのパッケージにDOS/V，PC9801，PC98XA/XL，PC98LT，PC286/386，FMR，FM TOWNS，FM16β，Panacom M，B16/B32，H2020，J3100/3300，AX，PS/55，IBM/PC，X68000の16機種用がおさめられている．

「FINAL II OS/2」（48,000円）
OS/2用のマルチファイルスクリーンエディタ．フリー・カーソル，ワンタッチ短縮入力など，ほかのFinalシリーズと統一されたテキスト編集環境を提供する．

「FOXY Ver.1.1」(予価48,000円)
MS-Windows 3.xx以上で稼働する日本語対応テキスト・サーバ．プログラム開発用スクリーンエディタの機能に加えて，DDE(Dynamic Data Exchange：動的なデータ交換）やOLE（Object Linkng Enbedding：アプリケーション間の連結機能）をサポートしているため，編集したテキストをほかのアプリケーションプログラムにダイナミックに引き渡すことができる．

「LE-LISP」「AIDA」「MASAI」(セット価格4,000,000円より)
仏ILOG社開発のX Window用GUI構築ツール．LISPインタプリタ／コンパイラLE-LISP，GUI部品ライブラリAIDA，画面上で対話型でGUIを作成するMASAIを利用することによって，LISPでアプリケーションを作成することができる．

ベル1も発表されている．すでにアメリカでは，MPC対応と銘打ったソフトも60種類ほど発売されているという．日本ではマルチメディアなどまだまだ遠い先の話のように感じられるが，海の向こうではソフト，ハードとも着々とマルチメディア化へ向けて動きが進んでいるといえそうだ．「日本でもFM TOWNSなどはCD-ROMドライブを搭載していますが，MPCのロゴをもらうのはいまのところ難しいでしょう．MPCでは，CD-ROMのデータ転送速度が1秒間150KBと定められています．これをクリアするのは技術的に結構難しいのです．日本のパソコンでは，VGAモード，256色というのもネックになるでしょうね」(小塚氏)．

また社内には，音楽スタジオがある．スタジオでは各種MIDI音源やシンセサイザーがあり，Macintoshの音楽ソフトを使ってデータの入力・編集などを行っている．将来的には，著作権の切れたクラッシックの名曲などをデータ化して，アプリケーションから利用したり，データ集として発売したりといった計画もあるのだそうだ．

●MPCレベル1仕様
CPU：80286（10MHz）以上
主記憶：2MB以上
表示能力：640×480ドット（16色または256色）
HDD：30MB以上
FDD：3.5インチ，1.44MB
CD-ROMドライブ：装備（データ転送150KB／秒以上，平均シーク時間1秒以下，CD-DA出力）
音声ボード：装備（WAVE，MIDI）
ジョイスティック：装備
マウス：装備

Windows MMEにも対応したテキストサーバーFOXY

Windowsのマルチメディア化をにらんで，エーエス・ピーがこの3月にも市場に送り出そうとしている最初の製品が，FOXYである．FOXYはFinalのWindows版ともいうべき製品で，Windows上でのテキスト編集のためのツールなのだが，単なるエディタではない．DDE，OLEというWindowsのアプリケーション間でのデータ受け渡しの機構を備えているため，ほかのWindowsアプリケーションを起動してテキストを編集する場合でも，テキストをクリックするだけで直ちにFOXYを起動し，編集するよう設定することもできる．編集した内容はそのままほかのアプリケーションにも反映される．また，FOXYでは各種のコマンドがボタン化されているので，ファイルのセーブやロード，カーソルの移動，編集したソースプログラムのコンパイルなどをボタンをクリックするだけで実行できる．また，Windows3.1版では，ボイス・ヘルプ，BGMなどの機能もサポートされる予定である．

FOXYは，現在発売されているWindowsアプリケーションと比較しても，Windowsのメリットを十分活かせるだけのスペックをもったソフトのように思えるが，小塚氏によれば，「日本ではまだWindowsプログラミングのための情報が十分ではない」のだそうだ．

「SDKに含まれるサンプルプログラムをいくら使ってみても，MS-WORD for Windowsのようなスピードを出すことはできません．まだ公開されていないAPIがあるのではないでしょうか．それに，せっかく労力と時間を割いて技術を身につけても，新しく3.1が出るとそれがすっかり陳腐化してしまうようでは困りますね．でも，パソコンでは，ようやくマイクロソフト社によるオープンシステムができあがりつつあります．基本的には，OSをのせてしまえば，ハードウェアはどこも大差なくなるのですから，ソフトウェアの側がしっかりと主導権を握ってもらいたいですね．そういった意味では，IBM互換機の市場には期待しています」．

ワンラインNEWS
米インテルが欧州の半導体市場で健闘し，テキサス・インスツルメンツを抜き4位のモトローラに肉薄し5位となった．トップは蘭フィリップス．

●イベント

■4月6日(月)～4月9日(木)
コミュニケーションTOKYO'92
「21世紀へのテレコミュニケーション」をテーマに，通信機器，通信システムを一堂に展示．
会場：東京国際見本市会場(東京・晴海)
入場料：無料
問い合わせ先：通信機械工業会　TEL 03-3231-3156

■4月8日(水)～4月10日(金)
CEPS JAPAN '92
コンピュータ，ワープロによる文書処理統合システムの展示，OSI，ODA，フォントテクノロジーなどに関するセミナーが行われる．「IMAGING '92」も同時開催される．
会場：サンシャインシティコンベンションセンターTOKYO(東京・池袋)
入場料：1,000円
問い合わせ先：カーナーズエクスポジションジャパン(株)　TEL 03-3349-8501

■4月14日(火)～4月17日(金)
DISK'92
光ディスク，磁気ディスクの製造技術，関連機器を展示．
DISK SYSTEMS'92
FDDやHDDなどのディスク装置を展示．
DISPLAY'92
液晶を中心としたフラットパネルディスプレイ装置の製造技術・関連機器を展示．
会場：東京国際貿易センター(東京・晴海)
入場料：1,500円
問い合わせ先：MESAGOジャパン　TEL 03-3359-0894

■4月22日(水)～4月25日(土)
マイクロコンピュータショウ'92
半導体，電子デバイスから応用機器，ソフトウェアシステムまでを含むマイクロコンピュータに関する総合的な展示会．
会場：東京流通センター大展示場(東京・平和島)
入場料：無料
問い合わせ先：(社)日本電子工業振興協会
TEL 03-3433-4547

■4月22日(水)～4月25日(土)
EDUCATION JAPAN
コンピュータや各種メディアを利用した学校教育，企業内・社会人教育のための機器を展示する「CAI&教育メディア・ショー'92」と「ミュージック・エデュケーション・ショー'92」，「キャリア・マネジメントフェア'92」が行われる．
会場：サンシャインシティ展示ホールA・B(東京・池袋)
入場料：500円
問い合わせ先：EDUCATION JAPAN運営事務局
TEL 03-3243-9082

●海外イベント

■4月6日(月)～4月9日(木)
COMDEX/Spring'92
コンピュータ，関連製品を一堂に展示．「WINDOWS WORLD '92」も同時開催される．
会場：McCormick Place(米国・イリノイ州)
問い合わせ先：ザ・インターフェイス・グループ
TEL 03-3237-8759

●セミナー

Microsoft University

■4月13日(月)～4月17日(金)
Microsoft Windows Applications Programming
Windowsアプリケーション開発のための上級コース．
受講料：180,000円(5日間)

■3月23日(月)～3月27日(金)
■4月6日(月)～4月10日(金)
Microsoft Windows Programming Environment
Windowsアプリケーション開発のための入門コース．
受講料：180,000円(5日間)

■4月13日(月)～4月14日(火)
C言語入門
Quick Cを使ったC言語の入門コース．
受講料：60,000円(2日間)

■4月13日(月)～4月17日(金)
MS OS/2 Presentation Manager Programming Environment
OS/2 PMアプリケーション開発のための入門コース．
受講料：180,000円(5日間)

■4月15日(水)～4月17日(金)
C言語応用
MS-C 6.0を使ってC言語の高度な技術を学ぶ実習中心のコース．
受講料：90,000円(3日間)

■4月21日(火)～4月24日(金)
MS OS/2 Programming Environment
OS/2 PMアプリケーション開発のための基本コース．
受講料：150,000円(4日間)

Microsoft Universityは，OS/2，Windowsアプリケーション開発者のために定期的に開催されているセミナーです．
問い合わせ先：富士ソフトウエア(株)
品川教育センター　TEL 03-5461-8296

NEWS & NEW PRODUCTS

パーソナルメディア，OOPも可能な統合開発環境を発売

パーソナルメディア（株）（TEL 03-5702-7858）は，C言語を用いたプログラム開発経験をもつプログラマであれば，約20語の予約語を学習するだけで，オブジェクト指向プログラミング，ルールベース，手続き記述の3つの異なるプログラミングスタイルを同時に利用することができる統合開発環境「RAL（ラル）」を発売した．「RAL」は，エキスパートシステム開発言語「OPS83」の開発者，C.L.Forgy博士が設立したProduction Systems Technologies社の開発した製品で，PC-9800シリーズをはじめとする各種MS-DOSマシンやサン・マイクロシステムズ社のSPARC-Station，ディジタルイクイップメント社のDEC-Stationなどのエンジニアリングワークステーション（EWS）で動作する．価格は，MS-DOSタイプが198,000円，EWSタイプが960,000円．また，TRON用OS「BTRON」にも移植される予定．

シンフォニー，AX対応のMicrosoft C Ver.6.0を発売

（株）シンフォニー（TEL 03-3443-0291）は，米国マイクロソフト社の開発した「Microsoft C Professional Development System Version 6.0」のAX対応版を発売した．アプリケーションプログラム開発の作業効率を向上させる「Programmer's WorkBench」，大規模なプログラムのデバッグが可能な「CodeView Debugger」，プログラムの構造をわかりやすく画面に表示する「PWBソースブラウザ」，ハイパーテキストベースのヘルプシステム「Microsoft Advisor」，JEGA，VGA/Hに対応した「グラフィクスライブラリ」などを備えている．価格は98,000円で，AX版「Microsoft C Ver.5.1」ユーザーには同社が優待販売を行う．

セイコーエプソン，Windows 3.0A，MS-DOS 5.0を発売

セイコーエプソン（株），エプソン販売（株）（エプソンPCインフォメーションセンター：TEL 03-3377-3531）は，同社の「エプソンPCシリーズ」用として，「日本語Microsoft Windows V3.0A」と，ユーザーメモリを大幅に拡大させ，Windowsライクな操作環境を実現した「日本語MS-DOS Ver.5.0」を発売した．「日本語Microsoft Windows V3.0A」は，386エンハンストモードでのレジューム機能への対応，イメージスキャナ入力ユーティリティの提供，ATOK7，MS-KANJI（VJE-βなど）の日本語FEPへの対応，日本向けに日付フォーマットを変更するなどの機能強化がなされた．価格は25,000円．「日本語MS-DOS Ver.5.0」は，従来のDOS機能と操作性を向上させ，大容量メモリを必要とするアプリケーションソフトを効率的に動作可能にし，ユーザーメモリの大幅な拡大，DOSシェルの搭載，データ復元やスクリーンエディタなどのユーティリティの追加により，機能，操作性を向上させた．価格は，基本セットが12,800円で，リファレンスマニュアルセットが4,000円．

富士通，拡張性を重視した16/32ビットノートブック型パソコンほかを発売

富士通（株）（FMインフォメーションサービス：TEL 03-3646-0816）は，拡張性に優れた32ビットノートブック型パソコン「FMR-50NBX2」，低価格な16ビットノートブック型パソコン「FMR-50NB2」，F9450シリーズとFMRシリーズの統合機「FMR-ΛII」を発売した．

「FMR-50NBX2」（写真左），「同50NB2」（写真右）は，新筐体の採用により，拡張性や携帯性などの使いやすさを高め，着脱式内蔵ハードディスクの採用，本体と一体型の拡張オプション（オプ

NEWS & NEW PRODUCTS

ションカード接続ユニット）により，省スペースでのLAN，ホスト接続が可能．従来機種にくらべ，約1.3倍（FMR-50NBX2），約1.5倍（同50NB2）の処理速度を実現，新FLバックライト付きLCDを採用している．

「FMR-ΔII」は，「FMR-50Δ」を機能強化，処理速度の向上，メモリ拡張をはかり，新筐体の採用で拡張性を強化した．価格は480,000円から730,000円．

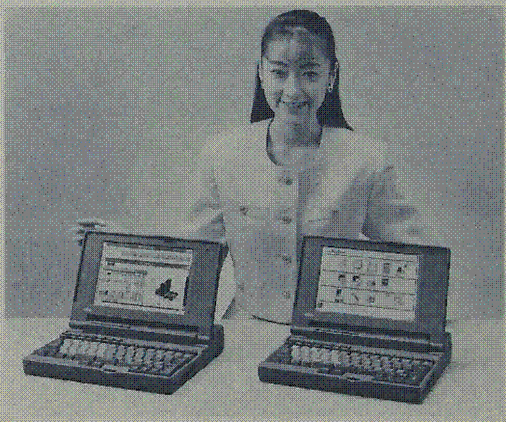
■FMR-50NBX2（FMR50NX21A/B，同26A/B，同27A/B）

プロセッサ：80386SX（20MHz）／メモリ：2MB（RAMディスク1.25MBを含む，最大10MBまで拡張可能）／外部記憶装置：3.5インチFDD×1・ICメモリカードスロット×1・40MB HDD×1（26A/Bのみ）・80MB HDD×1（27A/Bのみ）／ディスプレイ：モノクロ16階調FLバックライト付きLCD／価格：258,000円（21A/B），378,000円（26A/B），458,000円（27A/B）

■FMR-50NB2（FMR50NB21A/B）

プロセッサ：80C286（12MHz）／メモリ：2MB（RAMディスク1.25MBを含む，最大10MBまで拡張可能）／外部記憶装置：3.5インチFDD×1・ICメモリカードスロット×1／ディスプレイ：モノクロ16階調FLバックライト付きLCD／価格：198,000円

（※Aタイプはキーボード，Bタイプは親指シフトキーボード）



東芝，486，VGA対応カラーディスプレイ搭載ラップトップパソコンを発売

（株）東芝（パソコン営業推進部：TEL 03-3457-2930）は，ラップトップ型としてははじめて80486（33MHz）を搭載し，200MBの大容量ハードディスクを内蔵した「J-3100GT-XD」と，80486SX（25MHz）を搭載し，100MBのハードディスクを内蔵した「J-3100GT-XS」の2機種を発売した．両機種ともTFT方式10.5インチ大型カラー液晶ディスプレイを採用した「VTモデル」と，16階調プラズマディスプレイを採用した「Vモデル」の各2モデルがある．「日本語MS-DOS V3.1」を標準装備し，別売の「日英MS-DOS V5.0」（23,000円）により，DOS/V対応の日本語ソフト，IBM PC用の英語ソフト，IBM PC用VGA対応英語ソフトを利用することができる．

■J-3100GT-XD201VT/201V

プロセッサ：80486（33MHz）／メモリ：4MB（最大16MBまで拡張可能）／外部記憶装置：3.5インチFDD×1・200MB HDD×1／ディスプレイ：FLサイドライトTFTカラー液晶ディスプレイ（201VT），16階調プラズマディスプレイ（201V）／価格：1,398,000円（201VT），1,098,000円（201V）

■J-3100GT-XS101VT/101V

プロセッサ：80486SX（25MHz）／メモリ：4MB（最大16MBまで拡張可能）／外部記憶装置：3.5インチFDD×1・100MB HDD×1／ディスプレイ：FLサイドライトTFTカラー液晶ディスプレイ（101VT），16階調プラズマディスプレイ（101V）／価格：1,198,000円（101VT），898,000円（101V）

三菱電機，486搭載AXパソコンなど，MAXYシリーズの新モデルを発売

三菱電機（株）（情報処理機器事業部計画部：TEL 03-3218-3285）は，AXパソコン「MAXYシリーズ」の新シリーズとして，AX-VGA/Hに対応したデスクトップ型パソコン「DT3H/DT3」とラッ

NEWS & NEW PRODUCTS

プトップ型パソコン「LT4C/LT4」の各2モデル4機種を発売した。デスクトップ型は，80486SX，80386を搭載し，高機能・高性能化を実現，ラップトップ型は，80386SXを搭載し，ATフルサイズボードを内蔵することができる。TFTカラー液晶ディスプレイを搭載したカラーラップトップ「LT4C」とモノクロ16階調ディスプレイを搭載した「LT4」がある。

■DT3H（M3207H-A110/A117）

プロセッサ：80486SX（25MHz）／メモリ：3.6MB（最大63.6MBまで拡張可能）／外部記憶装置：3.5インチFDD×1・100MB HDD×1（A110），3.5インチFDD×1・170MB HDD×1（A117）／価格：1,158,000円（A110），1,318,000円（A117）

■DT3（M3207-A101/A110）

プロセッサ：80386（25MHz）／メモリ：3.6MB（最大15.6MBまで拡張可能）／外部記憶装置：3.5インチFDD×1・40MB HDD×1（A101），3.5インチFDD×1・100MB HDD×1（A110）／価格：798,000円（A104），958,000円（A110）

■LT4（M3202-C104/C110），LT4C（M3212-C104/C110）

プロセッサ：80386SX（20MHz）／メモリ1.6MB（最大13.6MBまで拡張可能）／外部記憶装置：3.5インチFDD×1・40MB HDD×1（C104），3.5インチFDD×1・100MB HDD×1（C110）／ディスプレイ：8/16階調モノクロLCD（LT4），TFTカラーLCD（LT4C）／価格：698,000円（M3202-C104），858,000円（M3202-C110），1,200,000円（M3212-C104），1,360,000円（M3212-C110）

マイタックジャパン，486，386SX搭載のOADG仕様パソコンを発売

（株）マイタックジャパン（TEL 03-5420-2822）はOADG仕様のデスクトップパソコンで，80486を搭載した「4060G」モデル，80386SXを搭載した「3060F」シリーズを発売した。それぞれすでに発売されている「4060F」，「3060E」の後継機で，各モデルの共通仕様として，4MBシステムメモリ，高速キャッシュメモリ，最高解像度1024×768のスーパーVGAディスプレイアダプタ，マウス，日本語106キーボード，日本語Windows高解像度ドライバなどを標準装備している。

■4060G（4060G-063-4/123-4/203-4）

プロセッサ：80486（33MHz）／メモリ：4MB（最大32MBまで拡張可能）／外部記憶装置：3.5インチFDD×1・60MB HDD×1（063-4），3.5インチFDD×1・120MBHDD×1（123-4），3.5インチFDD×1・200MB HDD×1（203−4）／価格：568,000円（063-4），618,000円（123-4），708,000円（203-4）

■3060F（063-4/123-4）

プロセッサ：80386SX（25MHz）／メモリ：4MB（最大16MBまで拡張可能）／外部記憶装置：3.5インチFDD×1・60MB HDD×1（063-4），3.5インチFDD×1・120MBHDD×1（123-4）／価格：358,000円（063-4），408,000円（123-4）

富士通，高速・大容量のLANサーバ専用機を発表

富士通（株）（サポートセンター：TEL 03-3730-8111）は，マルチベンダLANなど，パソコンのネットワーク利用の拡大に対応する，サーバ専用機「FMR-Serverシリーズ」2機種4タイプを発表した。同社のグループ企業でLANシステム事業で実績をもつ英国ICL社の製品に，富士通の技術を加えて商品化したもので，80486（33MHz），80486SX（20MHz）を搭載し，最大64MBのメモリ，最大3GBのハードディスクの搭載が可能。32ビットEISAバス，PC/AT互換アーキテクチャの採用，ディスクの二重化，無停電電源装置対応により，高速・大容量処理，高信頼性を実現した。ネットワークOSとして，「NetWare 386」，「日本語MS OS/2 LANマネージャ」を採用し，他社パソコンを含めたマルチベンダーLANシステムにも対応している。価格は，1,550,000円〜3,250,000円。発売は7月上旬の予定。

ワンラインNEWS
アイコンはロータス社の表計算ソフト「1-2-3」の印字機能を拡張するアドインソフト「1-2-3Publisher」を開発した。画面イメージどおりカラー印刷ができる。

NEWS & NEW PRODUCTS

アイ・オー・データ機器，標準ディスプレイで高解像度表示を可能にするオプションキット発売

（株）アイ・オー・データ機器（TEL 0762-60-3366）は，すでに発売している高解像度ディスプレイにより1024×768ドット，1677万色中256色同時表示が可能な高解像度グラフィックボード「GA-1024i」のオプションキット「GA-1024i用高密度表示オプションキット(GA-OP1)」を発売した．「GA-1024i」に装着することにより，640×400ドットの標準ディスプレイで，水平表示解像度アップと垂直表示スクロールの組み合わせ方式とインターレース表示，ノンインターレース表示の瞬時切り替え方式からなるNDHD(Normal Display Hi-Density)方式により，高解像度表示を実現する．価格は5,800円．

マイクロソフト，統合ソフトWorksの最新バージョンを発売

マイクロソフト（株）（インフォメーションセンター：TEL 03-3363-1201）は，PC-9800シリーズ対応の統合ソフト「Microsoft Works」の最新バージョン「同Ver.2.5」を発売した．「Microsoft Works」は，ワープロ，表計算，データベース，通信の４つの機能を１枚のフロッピーディスクにまとめ，すべてのツールの操作性を統一，各ツール間のデータのやりとりが自由に行える統合ソフトで，今回発売する「Ver.2.5」では，罫線機能のサポート，文字種類の拡張，スペルチェッカの装備などのワープロツールの強化，縦書き印刷のサポート，プリンタフォントへの対応などの印刷機能の拡充，すべてのツール上で一連のキー操作やかな漢字変換をマクロとして記録し，特定のキーにより再生するマクロ機能などの機能強化がなされた．価格は40,000円．また，２月28日から３月31日までの期間，「PC98NOTEカラー」などが当たるオープン懸賞を実施する．

ソフトウェアジャパン，日本語版ピーター・ノートン・バックアップを発売

（株）ソフトウェアジャパン（TEL 03-3862-2380）は，米国シマンテック社の「ピーター・ノートン・バックアップ」を日本語化し，発売する．複雑なバックアップ設定をせずに簡単にバックアップが行える初心者レベル，ユーザーの考えるバックアップイメージを簡単に実現，バックアップした内容をカタログファイルとして保存することのできる上級レベル，上級レベルで作成したカタログファイルを選択するだけで，複雑なバックアップを初心者でも簡単に操作することのできるプリセットの３種類のバックアップ機能を備え，任意にディレクトリを指定したリストア，各ハード機器のチェックや設定を詳細に指定することができる．PC-9800シリーズ対応で，価格は28,000円．

米国Micrografx社，日本法人を設立

Windows対応のグラフィックソフト「Micrografx Designer」などの開発元である米国Micrografx社は，100%出資の日本法人「マイクログラフィックス（株）」（TEL 03-5385-3130）を設立した．同社の製品には，Windows 3.0に対応し，DDE，OLE機能などを完全にサポートし，スライドショー，パッチプリント，ベジエ曲線での文字を含むあらゆるデータの編集か可能な「Micrografx Designer3.1」，スキャナで取り込んだイメージを補正したり，露出の変更をドットごとに編集，カラー写真のイメージデータを完全に加工編集することのできる「Micrografx Picture Publisher」などがある．なお，日本法人の設立にともない，従来のジスクソフト（株）との提携関係は解消される．

Programmer's Program
ソースレベル・マルチ・マクロ・デバッガ
PARTNER

PARTNERは，Cやアセンブラで開発したプログラムを効率的にデバッグするツールとして開発された．PARTNER ver. 3. 3は，従来のデバッグ機能を大きく進化させエディタ感覚でビジュアルなデバッグ環境と80286／80386プロテクトモード，EMSメモリ，ハードウェアブレークポイントのサポートによる先進的なデバッガである．

パッケージ

PARTNER ver. 3. 3はハードウェアブレーク機能をもたない80286以下のCPUのためにハードウェアブレークボードが付属したPARTNER-Hと，ソフトウェアのみのPARTNER-Sがある．今回試用したのはPARTNER-Sで，パッケージは250ページのマニュアル1冊とマスターディスク1枚，登録葉書から構成されている．マニュアルではインストールについてはあまり詳しく触れられていないが，マスターディスクのすべてのファイルをあらかじめ準備したディレクトリにコピーするだけである．

この中に含まれるPT. BATはPARTNERを起動するためのバッチファイルで，起動するデバッガの種類や各種オプションの例が書かれている．あらかじめ環境や対象コンパイラに合わせてエディタなどで変更する必要がある．とくに複数のコンパイラやアセンブラを使う場合は注意が必要である．

なおターゲットファイルのあるディレクトリがカレント以外の場合は，環境変数によってそのディレクトリをあらかじめ設定する必要がある．

デバッグのための準備

PARTNERでデバッグを行うには，ターゲットファイルにデバッグ情報が埋め込まれている必要がある．したがって，Cコンパイラあるいはアセンブラからデバッグ情報が正しく生成されていないと正常なデバッグを行うことはできない．

Microsoft C Professional Development System Version 6. 0の場合，まずコンパイル時に/Ziおよび/Odオプションを指定する．このオプションによって，デバッグ情報がオブジェクトファイルに書き込まれるとともにデバッグの妨げとなる最適化を禁止する．次にリンクである．付属のリンカで/COオプションを指定すると，CodeView用のデバッグ情報が実行ファイルに書き込まれる．つまり，MS-DOSのコマンドラインから

```
A> CL /Zi /Od ファイル名
A> LINK /CO ファイル名
```

などと入力する．

PARTNERの起動

PARTNERの起動は，MS-DOSのコマンドラインから

```
A> PT [<terget> <option>]
```

と指定する．tergetはターゲットファイル名，optionはターゲットファイルに対するオプションを指定する．PARTNERに対するオプションはPT. BATの中に記述するか，ターゲットファイル名の前に記述する．

PARTNERの起動画面には，ソースファイルの内容を表示するコードウィンドウと各種操作を行うためのコマンドウィンドウが表示される．PARTNERのおもな操作はこのコマンドウィンドウ内のプロンプトに対して，DOS同様のコマンドライン形式で行う．ほかにもショートカットキーやファンクションキーを用いて操作することができる．またコードウィンドウのオペレーション（表示，スクロール，カーソル移動，文字列検索など）はエディタ感覚で操作できる．

PARTNERの特徴

プロテクトモード，80386CPUのデバッグ機能，仮想86モードのサポート

PARTNERには，プロテクトモード版とリアルモード版が同梱されている．プロテクトモード版では，デバッ

ワンラインNEWS
富士通は専用回線を効率的に利用できる「LNモデム9600M」と専用線の障害時にISDN回線でバックアップする「F1979Mマルチポイント IBU装置」を開発した．

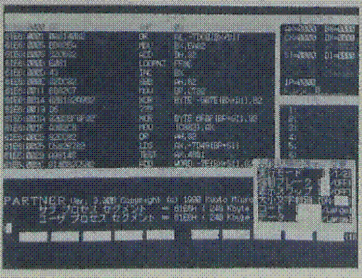
ガの本体が1MBより上位のメモリ空間にロードされプロテクトモードで動作するので，デバッガ自身がリアルモードのメモリ空間をほとんど使用しない．このため大規模なプログラムのデバッグにも利用可能．また80386以上のCPUが持つデバッグ機能や仮想86モードをサポートし，メモリやI/Oのアクセスによるブレークを外部ハードウェアなしに実現している．

各種メモリのサポート

デバッグ情報など，PARTNERのワーク領域をEMSあるいはプロテクトモード用の拡張メモリに確保することができる．これにより膨大なデバッグ情報をもつプログラムでもユーザーのデバッグ情報を制限することなくデバッグが可能になった．また，プロテクトモード版のPARTNERはDPMIおよびVCPIに対応しているので，これらの規格に準拠した仮想EMSドライバが組み込まれていても動作する．

ソースレベルデバッグおよび完全なデータタイプのサポート

Cやアセンブラのプログラムのデバッグをソースレベルで行える．さらに，Cのすべてのデータタイプおよび評価式をサポートしている．もちろんアセンブラベースのシンボリックデバッグも可能である（V30独自命令，8087/80287/80387のサポート）．



マクロ

PARTNERのコマンドラインから

＞{<マクロ名>

と入力すると，マクロ入力中を示すプロンプト「?」が表示され，マクロ入力モードとなり，[}]で再びコマンド入力に戻る．PARTNERのマクロコマンドには「DO..WHILE」などのループや「IF..」といった条件分岐，そして実行中のマクロの表示を抑制するコマンドが用意されている．

なおPARTNERのマクロ機能はバッファサイズに依存し，ステップ数に制限されることはない．この機能を使って，複数の内部コマンドを組み合わせて新しいコマンドを定義したり，ブレークポイントと組み合わせてデバッグ作業を効率的に行うことができる．

インディペンデントモード

PARTNERの起動時，-INDオプションを指定することによってインディペンデント（IND）モードとなり，MS-DOSから完全に独立してデバッグをすることが可能になる．つまり，INDモードではMS-DOSのシステムコールをいっさい使用しないので，MS-DOSのシステム内やデバイスドライバ内のルーチンにブレークポイントを設定することが可能になる．このモードによって，いままで困難とされてきたデバイスドライバのシンボリックデバッグが可能になる．ただし，このモードはソースレベルデバッグができないなどの制約があるので，通常のプログラム開発には向かない．

ほかにもカーソルによるデータ参照・変更できるインスペクト機能/デバッグ作業中にほかのMS-DOSコマンドを起動し，随時そのコマンドとPARTNERの間を切り替えることができるマルチジョブ機能/デバッグプログラムの実行に際して，どの関数がどれだけの時間を消費したのか関数別に計測表示するプロファイル機能/デバッグプログラムの実行に要した実時間を1ms（ハードウェアブレークボードがある場合には1μs）単位で計測するリアルタイムカウント機能など，豊富なデバッグ機能が用意されている．

■PARTNER ver.3.3

販売：京都マイクロコンピュータ（株）（TEL 075-953-0963）/価格：43,000円（PARTNER-S ver.3.3）/適応機種：日本電気PC-9801シリーズ，エプソンPC-286,386シリーズ（ハイレゾモードを含む）/対応コンパイラ・アセンブラ：MS-C（Ver.4.0/Ver.5.1/Ver.6.0），Turbo C（Ver.1.5/Ver.2.0），Turbo C++，Lattice C（Ver.3.x/Ver.4.1）/LSIC-86（Ver3.0），MASM（Ver.5.1），Turbo Assembler（Ver.1.0）/動作環境：MS-DOS Ver.2.11，3.1，3.3/必要メモリ：384Kバイト（プロテクトモード版はプロテクトメモリ512K以上）/供給メディア：5インチ2HDまたは3.5インチ2HD

オンラインソフト採集帳 16

鮪フォーマット対応のペイントツール 鮪ペイント

パソコン通信をしているといろいろなフォーマットの画像データを見かける．これらの画像データをディスプレイ上に再現するには，グラフィックローダーと呼ばれる画像データを再現するためのツールが必要となる．逆に自分でこれらのフォーマットの画像データを作るには，市販のペイントソフトを利用して絵を描き，該当の画像フォーマット用のグラフィックセーバを使って保存するのが一般的である．ところが鮪ペイントは直接「.MAG」を拡張子にもった「鮪フォーマット」の画像データを出力することができるペイントツールである．

鮪フォーマットの画像データとは

このツールは「鮪ペイント」という名前が示す通り，鮪フォーマットの画像データを作成するためのペイントツールである．鮪フォーマットとは，日本電気のPC-9801シリーズをおもなターゲットとして，草の根BBSである「まぐろBBS（旧まきちゃんネット）」で開発された（それ以前のフォーマットとして「MAKIフォーマット」が存在している），画像データフォーマットで，拡張子に「.MAG」をもっている．拡張子の「MAG」は「MAkichan Graphic」の略で，そのフォーマットの画像データファイルをグラフィックVRAMにロードするためのツールである「MAGローダー」が「鮪だ」，「MAGフォーマット」が「鮪フォーマット」と呼ばれている．

このフォーマットはアナログ16色で画面サイズに制限がなく，圧縮を行いファイルサイズを小さくしているのが特徴．ただし，単純な圧縮アルゴリズムのためかイメージスキャナで読み込んだ写真のようにディザの多い画像データには向いておらず，おもにアニメのセル画などのようなベタ部分が多い画像に利用されている．

鮪ペイントの起動

まず環境変数の設定を行う．鮪ペイントが参照する環境変数はUSERとMPSTMPの２つである．鮪フォーマットの画像データには作者名も記録されるので，USERは画像データの作者名を指定する．MPSTMPには作業用ドライブの指定をするので，アクセス速度の比較的高速なRAMディスクなどを指定する．ただしMPSTMPが設定されていない場合は，PSTTMP，TMPの順に参照し，どれも設定されていないときはカレントディレクトリのルートに設定される．AUTOEXEC.BAT中に

```
set USER=yoshi
set MPSTMP=H:¥
```

などと追加しておけば，起動時に自動的に設定される．起動はMS-DOSのコマンドラインから

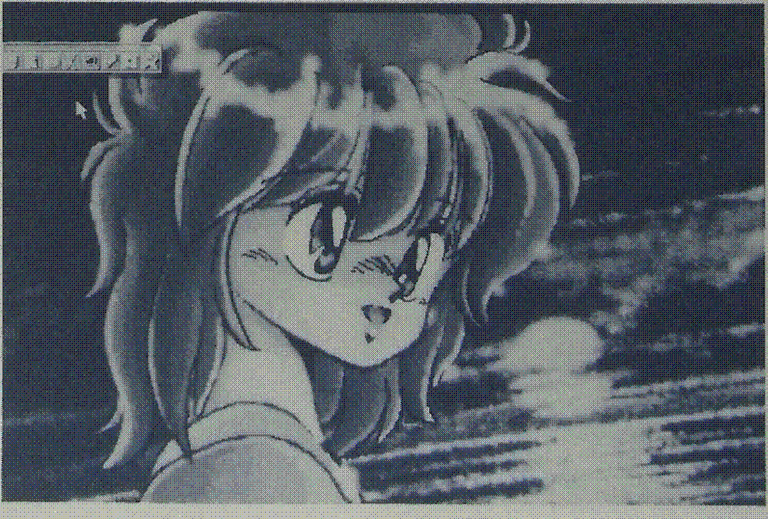
```
A> MPS
```

と入力するだけである．

なお鮪ペイントはマウスドライバを内蔵しているので，新たに組み込む必要はない．またドライバがすでに組み込まれている場合は，組み込まれているマウスドライバを利用する．対応するマウスドライバはMicrosoft Mouse互換のドライバでMS-DOSに付属するNECのマウスドライバには対応していない．また起動オプションで-mを指定すれば内蔵しているマウスを強制的に使用することもできる．

環境ファイルの設定

環境ファイルMPS.CFGには，最大10の外部ローダーや，最大30までのチャイルドメニューによる外部コマンド



ワンラインNEWS
シャープは3.5インチのFDDを2基搭載した省スペースWS「X68000コンパクトXVI」を発売．従来機に比べキーボードもコンパクト化した．価格は29万8千円．

を登録することができる．各画像フォーマット用のグラフィックローダーを登録しておけば，ファイルウィンドウから直接読み出すことができるようになる（ただしセーブすることはできない）．また外部コマンドも鮪ペイント用のツールが数多く公開されているので，これらを登録すればさらに便利になるだろう．ほかにマウスの感度を設定することもできる．

各ツールの説明

鮪ペイントが起動すると現在のグラフィック画面を表示し，左端に小さなウィンドウ（＝メインメニュー）が表れる．メインメニューには8つの図形が表示され，クリックすることで左側からそれぞれペンウィンドウ，カラーウィンドウ，マスクウィンドウ，エディットウィンドウ，ルーペウィンドウ，パレットウィンドウ，ファイルウィンドウ，システムウィンドウを呼び出すことができる．タイトルバーまたはメインメニューの該当する図形をマウスの右ボタンでクリックすれば，それぞれのウィンドウを消すことができる．またルーペウィンドウのみウィンドウサイズが可変となっており，右下のリサイズボタンをドラッグすることで任意の大きさにすることができる．

各ツールの使い方は付属するドキュメントファイルに詳しく述べられているが，実際に試してみるのが早いだろう．

はじめて鮪ペイントを使うときには，機能の多さから操作が難しく思えるかもしれない．ところが，その機能が便利であるため，つい使い続けているうちに慣れてしまうようだ．ただし，うまく描けるようになるかどうかは使う人の絵心次第？

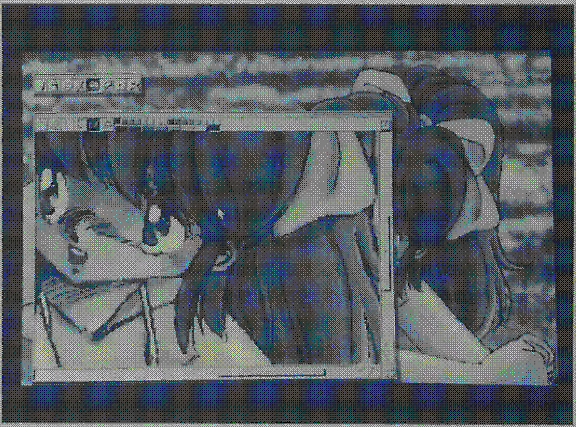
■鮪ペイント Ver.0.18

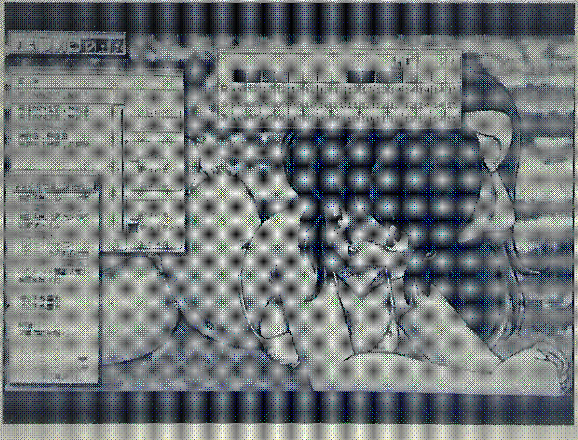
名称：MPS.EXE／分類：ペイント／登録名：MPS018.LZH／作者：W RINN／動作環境：PC-9801シリーズ（VM以降，ただしVM,VFは16色グラフィックボードが必要）／登録月日：1991年12月23日／復元方法：LHA E MPS018／入手場所：ASCII-NET PCS（pool msdos #4806）

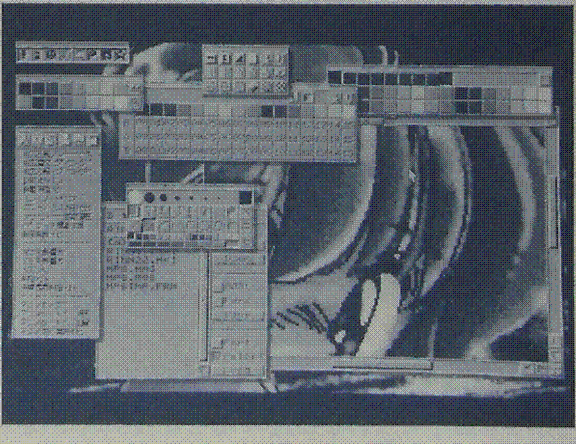
ファイル構成

Listing of archive : MPS018.LZH

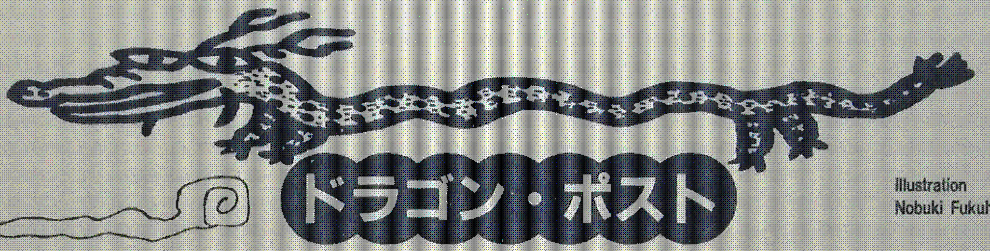
Name	Original	Packed	Ratio	Date	Time	Attr	Type	CRC
MPS.DOC	45888	18595	40.5%	91-12-20	04:06:34	a--w	-lh5-	78C9
MPS.EXE	126048	54658	43.4%	91-12-20	03:02:20	a--w	-lh5-	82D0
MPS.MAG	3698	2214	59.9%	91-07-23	00:12:24	a--w	-lh5-	7CD6
MPS.CFG	1038	497	47.9%	91-12-20	04:11:44	a--w	-lh5-	6542
4 files	176672	75964	43.0%	91-12-23	20:37:26			







アップルコンピュータは，同社の「Macintosh」シリーズ向けの日本語アウトラインフォントソフト「漢字TrueType」を開発した．



ドラゴン・ポスト

Illustration
Nobuki Fukuhara

この欄は，読者からのお便りを中心に構成しています．本誌に対するご意見・ご感想などをお待ちしております．鬼より恐い読者からの声ということで，「ドラゴン・ポスト」です．本誌についているアンケートはがきにご意見の欄を設けてありますから，そこに書いてください．ペンネームでも歓迎です．ただし，はがきには本名と住所をしっかり書いてくださいね．掲載誌をお届けします．

★もうパソコン少年じゃない……

わたくしパソコン少年からプロ（パソコンじゃないんですが）になった者ですが，先日，親せきの子から「パソコンを買おうと思うんだけど，どんなのがいいか教えて」と言われ，近ごろ仕事がいそがしく，時価はもちろん，最新機種なぞ知りもしない，という事に気付きガクゼンとしました．好きなコンピュータをさわってはいても，もうパソコン少年じゃないんだなぁ，と．イイヤ，覚えきれないくらいの同系マシンを作ったボウメーカーが悪い！番号変えればいいのに，98，99，100……（細川哲哉）

――次々に発表される新製品をぬかりなくチェックするには相当なパワーが必要です．それも，少しでも安く買おうというくらいの真剣味がないと，実売価格までチェックするのは難しいものですね．とくに買った直後に新製品が出たりすると，気になりつつも知りたくなかったりして……．

★TOWNSでプログラミング

今，TOWNSをよく使用しています．やはり32bitのプロテクトモードが標準なのはいいですね．しかし，まだまだプログラミング関係のものが少ないと思います．個人的にはボーランドの物が気に入っているので，BORLAND C++のプロテクト対応モードつきをCDで出してもらいたいです．あのコンパイルスピードにはまいった．HD上やRAMDISK上だとめちゃはやい．ほしー．（柿下卓）

――私もDOS/V版のBorland C++ 3.0アプリケーションフレームワークが欲しい！　ボーランドさんお願い，早く出して！^^;

★計画は万全だったのに

SASI40メガのHDDを友人に譲った．もちろんすぐに100メガ以上のSCSI HDDを買うつもりだったのだが，3.5 inchの光磁気ディスクもいいかもしれないと考えていた．ところが生活費に困ってしまって，その金を使ってしまった（なぜ困ったか自分でもわからない．計画は万全だったはずなのだが……）．おかげで今，何もできない．FDDオンリーでは苦しすぎる……（自業自得とも言う）．（Fake Maker）

――そうなんですよ，何度計算してもバッチリうまくいくはずなのに，なぜか生活が困窮してくるんですよ．でも，やっぱりコンピュータを買うのはやめられない……．

★消されたID

SE-NETのIDを2度もらい2度も消えた．回線が少ないので最近はつながりません！　BBSKIDの名も消えたSE-NET？　3度目のIDがほしいよー．再登録できるまでチャレンジ！　IDがもらえたらまたハガキを出します．（BBSKIDよろしく！）

――回線が少なくてつながらないという苦情の多いSE-NETですが，読者の皆さんとシスオペの努力でなんとか続いているようです．3度目の登録をしたら，こんどはSE-NETでお会いしましょう．:-)

★『情熱』が大切

モジュール設計して，分担してコーディング，統合テスト，試験……みたいにして出来上がった市販ソフト．たった一人の天才が趣味で作ってフリーウェアにしたソフト．私は後者を使うことが多い．決して「タダだから」ではなく（どーせ会社の金:-P），機能や使い易さで選んだ結果である．これを私なりに分析してみると，「一人で」「天才が」「趣味で」作る場合，必然的にそのソフトには『情熱』がそそがれるからだと思う．というわけで，前者ソフトを作る立場の私も情熱を大切にしようと考える今日この頃である（コピープロテクト方法を考える情熱はない）．（mog）

――そうです，そのとおりだと思います．エンジニアリングだけでは解決できない，プログラミングのアートの部分に注目したいですね．

★無駄使いが好き!?

アルバイトが忙しいので，日に2～3時間しかパソコンで遊んでいません．WXⅡ＋を買っても，めんどくさいのでインストールしてなく，EMSボードメモリを買ってもシステム以外は逃がしてありません．MIDIを買って，サンプル曲ばかり聴いています．なんだか，お金の無駄使いが好きになってしまいました．（松村恵一）

――そうして，どんどんハマッていってしまうんですよ．

気がつけば，部屋には周辺機器やらソフトの箱やらがゾロゾロと……^^;

★見捨てないでね

アプリプログラマ志望のぼくはNS/EとGE3のどちらをかえばいいんでしょう？　30万円で一式そろえるのは辛いものがあります．でも実際に動かさないと実力がつかないし，16ビットポケコンでがまんしようかなぁ．でも使えない付録ディスクがたまっていく．当分ペーパープログラマからぬけだせません．こんなぼくでもみすてないでね．（山本智章）

――秋葉原のあるショップでは，近々19万8千円でフルセットのDOS/Vマシンを売り出すとか？　もう少し待ってみませんか．

★Macの通信ソフトがいっぱい

通信ソフトの話はおもしろかったけど，98で通信を始めようとしたら大変ですね．私は98用は，あの古い「蘭98」しかもってないけど，片やMacは，PDSでBMUGから，Free Term，ZTerm，Ninja Termなどの通信ソフト，Micro Phone IIなどの市販品最強のソフトのスクリプト，それにあまつさえHermes BBA，World WarIVなどのホストプログラムまで各4ドルで手に入ってしまう．おかげで，私はモデムもないのにソフトだらけ……．（佐伯俊也）

――98にもIBM PC互換機にもフリーの通信ソフトやホストプログラムはたくさんあります．98用ならWTERMがいいと思いますよ．

★ブラインドタッチ練習中

私は今キーボード入力をブラインドタッチで行えるようにがんばっていますが，プログラム開発にたずさわっている方々はどうされてるんでしょうか？　疲れ方や入力時間に違いが大きく出てくるのでしょうか．そのような問題も少し取り上げていただきたいと考えます（タイプ練習用のソフトもあるくらいですからやっぱりブラインドタッチかな？）．（川端政行）

――プログラマできちんとしたタッチタイピングを身に付けている人は，むしろ少ないんじゃないでしょうか？タイプスピードは早くても，打ち方は自己流の人が多いようです．

★ハードディスクが欲しい！

ハードディスクが欲しい！　と思ってお金を一生懸命ためています．私，高校生にとって毎月のこづかい5,000円のうち3,000円を貯金して，2,000円しか使えないのはとても苦しい．まだ高校生になってから貯金しはじめて2万円しかたまっておりません．将来のことを考えてぜったい100MB以上……．あと何年かかるでしょうか？（中島大輔）

――ちゃんと貯金しているなんてエライ！　ハードディスクもどんどん値下がりしていますから，あともう一息．掲載号をお送りしますので，今月は3,980円貯金してくださいね．

★プロのプログラマはたいへん！

とあるゲームソフト会社で，アルバイトとしてグラフィックツールを作っているのですが，職業プログラマとなるとけっこう大変だなー．と実感しています．趣味でやっているときとちがって，「いつまでにこれだけのことをやらなきゃならない」わけで，この締め切りとノルマと義務感とで頭がいたい今日このごろであります．（どくきのこ）

――プロとアマチュアの違いは，納期を守れるか守れないかだといいます．プロとアマチュアのライターの違いは，……いわないほうがいいですね．^^;

★先端技術とトレンドを追っかけて！

FA関係のソフト開発をしています．これまでは言語としては10以上使ってきましたが，現在はCがほとんどです．興味があるのはOOPとWinです．貴誌ではよく言語の比較記事がありますが，それほど意味や要望があるのでしょうか？　どの言語を使うかって議論するなんて非生産的じゃないですか？　特集では先端技術とトレンドを追っかけて欲しい！

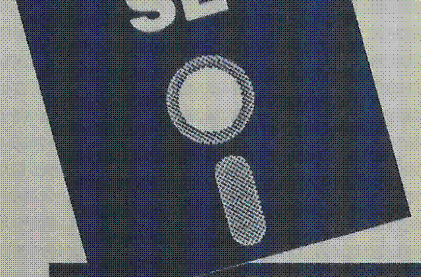
――作るプログラムに合わせて言語を変えるのは，悪いことではないと思います．また，さまざまな言語を知ることで，よりプログラミングに関する知識が深まるのもほんとうですよ．

★たのしく，理解しやすい「イカB」

「イカすBASICプログラミング入門」の酒井氏の文章に大変感心をしました．手続き型言語を説明するための“名義変更”や“IF文”の「大変」理解しやすい処理内容の例文はたのしく，また理解しやすくなる文章でした．職場で人に説明することが多いので，酒井氏のような説明ができたらなと思いました．「人に説明するためのテクニック」なんていうテーマの特集はどんなものでしょう．（青木与志夫）

――街にフレッシュマンの溢れる季節にふさわしく，来月号の特集では「Cの教え方のコツ」を掲載する予定です．

（敬称略）



付録ディスクのご案内

このフロッピーディスクに含まれる実行ファイル，ソースリストおよびテキストファイルは「月刊Programmer's Page」1992年4月号に掲載されたものです．

付録のフロッピーディスクは，5"2HD（8セクタ／トラックMS-DOS）でフォーマットされています．3.5"や2DDなどへのコンバートサービスは行っておりません．なお，オンラインサービスSE-NET（03-5276-1297）でも，同じ内容のファイルを用意しています．

フロッピーディスクの内容は次の通りです．本誌およびディスクに含まれるドキュメントファイルをよく読んでからご利用ください．

■ディスクのおもな内容

●付録ディスクについての説明

ディレクトリ名：ルート

収録ファイル名：

README.BAT（ディスク内容の説明を読むためのバッチファイル，PC-9801専用）

README.DOC（ディスク内容の説明，MS-DOSテキストファイル）

ONLINE.BAT（オンラインソフトの説明を読むためのバッチファイル，PC-9801専用）

ONLINE.DOC（収録されているオンラインソフトの簡単な説明）

V.EXE（ミニページャー，PC-9801専用）

BOOK.TXT（翔泳社書籍目録）

●特集「新しいDOS環境を探検する」

④「DOS/Vとはなにか？」（p.30）

ディレクトリ名：TOKU

収録ファイル名：TOKU4.LZH

⑥「MASM6.0の機能と特徴」（p.54）

ディレクトリ名：TOKU

収録ファイル名：TOKU6.LZH

●プログラム「遊」工房（p.78）

フロッピーディスクの直接読み書き「trackrw」

ディレクトリ名：PLAY

収録ファイル名：PLAY.LZH

●実践C塾　アルゴリズム編（p.90）

サーチ・その2

ディレクトリ名：JUKU

収録ファイル名：JUKU.LZH

●98活用アセンブラ技法　ブロックデバイス編（p.100）

不揮発性RAMDISKドライバの作成

ディレクトリ名：98ASM

収録ファイル名：98ASM.LZH

●PRODUCTS REVIEW（p.113）

Microsoft FORTRAN Professional Development System Version 5.1

ディレクトリ名：REVIEW

収録ファイル名：FORTRAN.LZH

●綴じ込み付録

ディレクトリ名：DATA

収録ファイル名：MASM60.TXT（MS-DOSテキストファイル）

●オンラインソフト採集帳（p.136）

92年1月号のオンラインソフト採集帳で紹介した高速テキストページャー「MIEL」の最新版と，1つまたは複数のファイルの圧縮および「.LZH」という拡張子をもった圧縮ファイルの復元をする高圧縮復元書庫管理ソフト「LHA」が収録されています．なおMIELはLHAによって，LHAは自己復元型で，それぞれ圧縮されています．復元方法についてはONLINE.DOCを参照してください．

なお本誌今月号のオンラインソフト採集帳で紹介した「鮪ペイント」は都合により収録されておりません．

ディレクトリ名：ONLINE

収録ファイル名：

MIEL241.LZH（MIEL v.2.41，PC-9801用）

LHA213.EXE（LHA v.2.13，MS-DOS汎用）

●クイズ・ランダムスクリーン（p.150）

クイズ解読ツール「PUZLE」

本誌掲載のクイズ・ランダムスクリーンの画像をディスプレイ上に再現し，マウスを使って並べ換えることにより，解答を得ることができるユーティリティです．PC-9801シリーズ専用です．詳しい使い方はPUZLE.DOCを参照してください．

ディレクトリ名：OMAKE

収録ファイル名：

PUZLE.EXE（解読ツール，PC-9801専用）

PUZLE.DOC（PUZLEの説明）

CONFIG.SYS（PUZLE用CONFIG.SYS）

AUTOEXEC.BAT（PUZLE用AUTOEXEC.BAT）

RNDSCREN.LZH（画像データ）

TULIP.LZH（3月号の解答データ）

TULIP.BMP（Windows 3.0の壁紙）

※使用上の注意

本誌に掲載されたプログラムおよびデータを利用するには，MS-DOSの基礎知識が必要です．このディスクには，MS-DOSのシステムは入っていません．別途，MS-DOSが必要となります．また，ソースファイルのコンパイルなど，別途コンパイラやユーティリティが必要な場合があります．

■圧縮ファイルについて

多くのファイルは圧縮した上で収録しています．「.LZH」という拡張子のついたファイルは，高圧縮書庫管理ユーティリティ「LHA」によって圧縮されたファイルです．これらのファイルを利用するには，まずLHAで復元する必要があります．LHAの利用方法については，ONLINE.DOCをご覧ください．

プロトタイプツール
「PROTO TYPIST」デモ版
（PC-9800シリーズ用）
Cドキュメント作成支援ツール
「MICRO DOC」デモ版
（PC-9800シリーズ用）

サブディレクトリONLINEに，マイクロアート㈱のプロトタイプツール「PROTO TYPIST」とCドキュメント作成支援ツール「MICRO DOC」のデモ版を収録しています。「MICRO ART.LZH」がデモ版をディレクトリごとの圧縮したファイルです．復元するディスクの空き容量が1.1Mバイト以上必要です．復元方法についてはONLINE.DOCを，使用方法については復元されたディスク中のREAD.MEを参照してください．

協力：マイクロアート株式会社

バーチャルスペースガイド

P子　某ソフトハウスで企画営業を担当している．26歳．
C平　某メーカーのシステムソフトウェア開発部門に所属．25歳．

C平　出たんだってね．
P子　何が？出ただけじゃわかんないよ．
C平　何がって，「ターミネータ２」の話．ビデオとLDで３月６日にリリースされたの．これで，あのCGを使った特撮場面もじっくり見られるわけ．
P子　ふーん．じっくりとなると，やはりLDだろうね．静止させたり，コマ送りにしたり，AVオタクとしはこたえられないだろうね．
C平　だれがAVオタクだっていうの？今度のは，バージョンが多いんだ．テレビ用にトリミングした版と劇場公開シネスコサイズ，それにテレビ版には吹き替えもある．僕としては，29インチの大型画面でシネスコサイズをたのしんじゃう．
P子　いまどき29インチは大画面とはいわないような気もするけどね．でも，そんなにすごいのあの映画のCG？
C平　劇場公開されたときから，CGの最先端技術を使っているといわれていて，液状金属でできたサイボーグが変化していくところをデジタル画像処理で作っているんだ．なんでも，デジタイズしたグラフィックをいくつも取り込んでおいて，それぞれを結ぶ動きはデジタル処理をするらしいんだけど．CGをやっている友達に聞いたら，あれは最先端の技術だっていっていた．
P子　なるほど．最先端技術をお固い分野じゃなくエンターテイメントに使ってしまうところが，またすごいわけだね．

■ターミネーター2

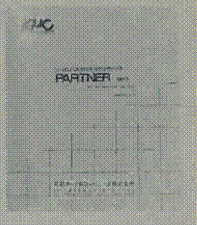
(1991年，米)
監督：ジェームス・キャメロン
出演:アーノルド・キュワルツェネッガー
5,700円　パイオニアLDC

機械と人間が対決する未来社会をバックにして，タイムトラベルもからんだ戦闘を繰り広げるシリーズ第２弾．今回は未来社会から送り込まれた新サイボーグのCG映像が見もの．シュワちゃんは前作から一転して正義の見方として活躍する．

読者プレゼント

応募は本誌に添付のアンケートはがきをご利用ください．(1992年４月17日消印有効)．なお，応募者多数の場合は，抽選になります．当選者発表は発送をもって代えさせていただきます．

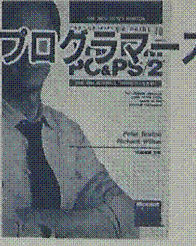
1 PARTNER Ver.3
(PC-9800シリーズ用)



提供：京都マイクロコンピュータ㈱

1名

2 IBM PC&PS/2プログラマーズガイド

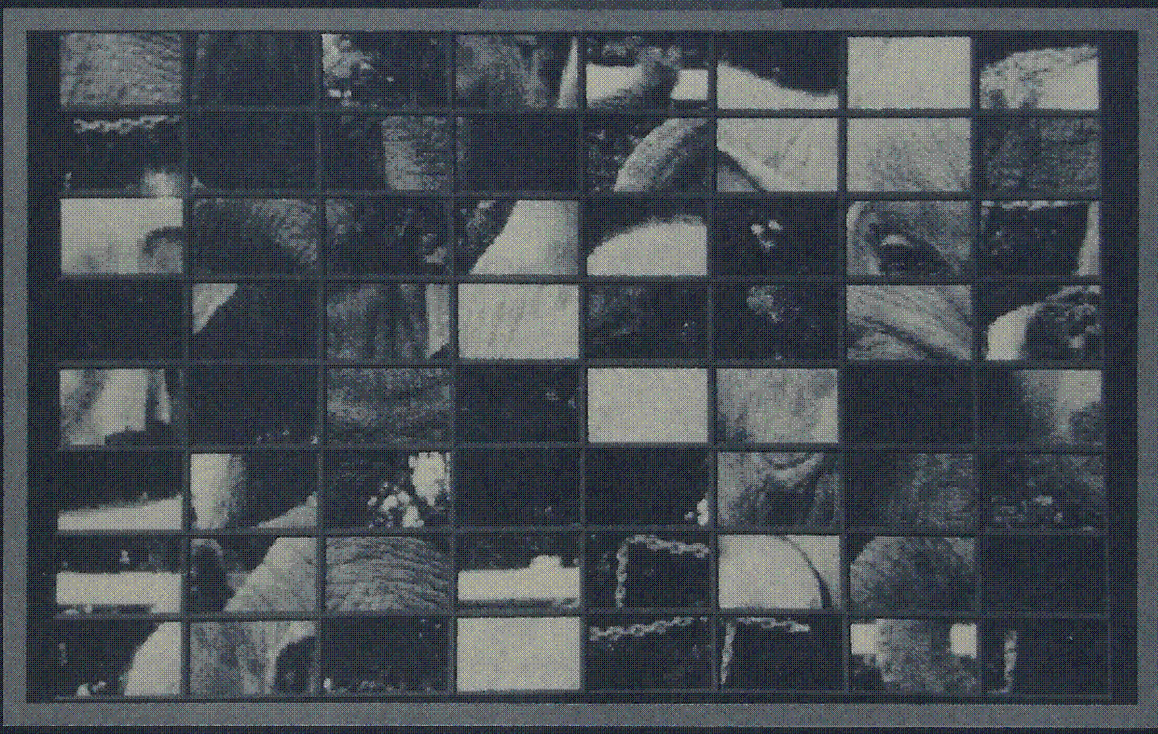


3名

クイズ・ランダムスクリーン

ある１枚の画像を64分割し，ランダムに並べ替えてあります．
さて，もとの画像は何を写したものでしょうか．

ヒント：インドやアフリカにいます



正解者の中から30名様に本誌表紙イラスト（日暮修一画）入り月刊プログラマーズ・ページ特製オリジナルマグカップをさしあげます．

応募方法：本誌に添付のアンケートはがきをご利用ください．

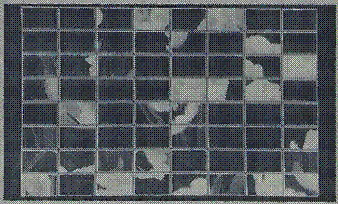
締め切り：1992年４月17日（消印有効）

なお，当選者の発表は発送をもって代えさせていただきます．

＊画像データは，付録ディスクやBBSでもサービスしています．

３月号の問題と正解

【問題】



【正解：チューリップ】

SE-NETからのお知らせ

SE-NETでは開設以来，オンラインサービスを行ってきましたが，回線混雑のため非常に接続しにくい等の問題が出てまいりましたので，GUESTによるダウンロードは中止させていただきます．今後は読者のみなさまと編集部とのコミュニケーションの場としてご利用ください．

●BBSへの接続

SE-NET　03-5276-1297

通信速度	300/1200/2400bps
接続プロトコル	CCITT V.22bis/V.22/V.21
エラー訂正プロトコル	CCITT V.42，MNPクラス５
漢字コード	シフトJIS

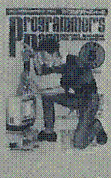
Programmer's Page
MONTHLY MAGAZINE FOR PROGRAMMERS

月刊プログラマーズ・ページ バックナンバーのご案内

■定価・各980円

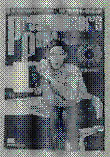
1991年5月号 5インチFD付き
特集・基礎力としてのアセンブラ

プリミティブなプログラミング言語でありながら，それ故に無視することができないアセンブラ.コンピュータを理解し，その能力をひきだすためにもう一度アセンブラをみなおす.



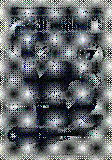
1991年6月号 5インチFD付き
特集・研究「プログラミング統合環境」

今流行のエディタ，コンパイラ，リンカ，デバッガを統合した開発環境について，その歴史的系譜から最新のシステムまで徹底的に検証する.



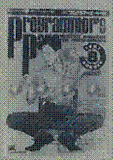
1991年7月号 5インチFD付き
特集・最新「デバイスドライバ」研究

パソコンの拡張性を支えるソフトウェアインターフェースであるデバイスドライバ作成の技法と特徴をMS-DOS, Windows, OS/2の各環境ごとに解説する.



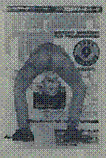
1991年8月号 5インチFD付き
特集・はじめての
オブジェクト指向プログラミング

"オブジェクト指向"の歴史的な発展の過程から基礎概念，分析・設計の方法までを網羅的に解説する.



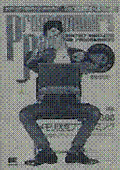
1991年9月号 5インチFD付き
特集・グラフィックス
プログラミングの世界

基礎からフレームバッファの活用，レイトレーシング，Windows，IBM PCのグラフィックスなど，その技法を徹底的に解説する.



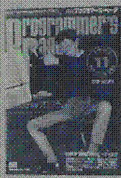
1991年10月号 5インチFD付き
特集・286/386
メモリ活用プログラミング

バンク式メモリ，EMS，XMSからプロテクトモードまで，640KBの壁を破るための数多くのプログラミングテクニックを解説する.



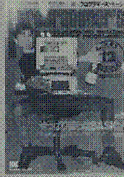
1991年11月号 5インチFD付き
特集・Windowsプログラムの作り方

正統派MS-C＋SDKをはじめ，VisualBasic，BC++，TPWなどの新しい開発環境，ツールによるWindowsプログラムの作り方を探る.



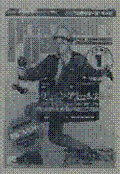
1991年12月号 5インチFD付き
特集・やっぱり僕らはBASIC/

構造化プログラミング，マルチランゲージに対応した強力なプログラミング言語として生まれ変わった現在の進んだBASIC，MS-BASIC 7.1，TrueBASICからVisualBasicまでを紹介する.



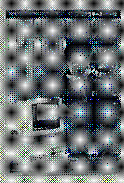
1992年1月号 5インチFD付き
特集・プログラミングツール大集合

プログラム開発，メンテナンスに欠くことのできないプログラミングツール，その利用法から，使いこなしのコツまでを紹介する.



1992年2月号 5インチFD付き
特集・通信ソフトの作り方

RS-232Cを使用した通信の概要を解説し，ライブラリ，ライブラリを用いた通信用マクロ言語を作成，BBSなどにアクセス可能な通信プログラムを完成する.



1992年3月号 5インチFD付き
特集・Cじゃなくてもいいじゃない

C言語の実体をとらえなおし，PascalやMODULA-2など，ほかのプログラミング言語と比較することでプログラミング言語の能力と性能を解説する.

1991年1月号「特集・プログラマの仕事場」・2月号「特集・Windowsの時代である!」・3月号「特集・この言語を使いなさい!!」・4月号「特集・C読本」は品切れです

「月刊プログラマーズ・ページ」のバックナンバーは，ご注文いただければ，全国どこの書店でもお求めになれます(書店に届くまで1か月ほどかかります)．小社に直接ご注文なさるときは，何月号かを明記の上，現金書留にて雑誌価格＋送料を右記までお送りください．送料は1部81円です（ただし7月号は1部86円）．2冊以上の場合は，号数・冊数により送料が異なりますので，小社までお問い合わせください．

〒102 東京都千代田区平河町
2-4-14 平河町KSビル4階
(株) 翔泳社
マーケティング部PP係宛
TEL 03-3263-0447

Programmer's Page

MONTHLY MAGAZINE FOR PROGRAMMERS

月刊プログラマーズ・ページ5月号

(4月18日発売)

付録：5″ 2HDフロッピーディスク付き!!

特集

Cの学び方・教え方

Cがわかる，うまくなる学習方法を探る

パソコン用開発言語としてもっとも多く使用されているC言語について，その学び方から教え方まで，プログラミング学習のよりよい方法を模索する．また，MS-C，BC++の2大処理系の互換ライブラリの作成を試みる．

月刊プログラマーズ・ページ　1992年4月号
1992年4月1日発行

発行所　株式会社翔泳社
東京都千代田区平河町2-4-14平河町KSビル4F・5F
TEL 03-3263-0447　〒102
発行人　長廻健太郎
編集長　清水　隆
副編集長　鈴木嘉平
編集スタッフ　杉野貴美子＋上田純美礼＋芳山充弘
表紙デザイン　SE（翔泳社デザイン研究所）
広告スタッフ　相澤正夫＋澤本良知＋種村嘉彦
協力　根本裕三郎＋(株)ドモン・マインズ＋佐藤裕美＋保坂庸介＋小林弘人(図版デザイン)
写植　株式会社東巧社＋株式会社トーヨー・プリント社
印刷　凸版印刷株式会社
定価　980円（本体951円，税29円）

© 1992 SHOEISHA
本書の一部または全部を著作権法の定める範囲をこえ，無断で複写，転載，テープ化，データファイル化することを禁じます．
＊本書の内容に関するご質問は，ご面倒でも必ず文書にて編集部までお問い合わせください．
＊本誌に記載した会社名，ハードウェア・ソフトウェアの商品名は，各社の商標または登録商標です．

EDITOR'S PAGE

●Windowsのようなグラフィックベースの環境が普及し，それがアウトラインフォントに対応すると，画面も印刷も美しい文字で表現できる，というのは英語の話．日本語ではかなりの困難が予想される．なにしろアルファベットと漢字では，文字の数が圧倒的に違う．フォントファイルは，たいへん大きなものになる．英語のようにプルダウンメニューに表示された中から好きなフォントを使うわけにはいかないだろう．現在主流である16ドットフォントが判別できる漢字を表示するのに精いっぱいなことを考えると，まだハードウェアの制約が大きい．とりあえずは，日本語Windows3.1に付属すると伝えられる日本語のTrueTypeフォントがどのようなものか期待が持たれる．それにつけても，画面の狭さ，画数の多さ……．(TA)

●最近，PC用に開発されているOSのリリースが大幅に遅れている．ソフトウェアの開発が予定どおりに進まないのは珍しいことではないが，どれもこれもとなると，ちょっと気がかりである．思わず，ソフトウェアクライシスなんていう，なつかしい言葉を思い出してしまったりする．そのような中で，ボーランドだけが“ユーザー泣かせ”といわれるほどの矢継ぎ早のバージョンアップ，ニュープロダクツのリリースを行っている．こうなると，“オブジェクト指向の有効性”というのもあながち嘘ではないように思えるのだが……(嘉平)

●「カルトQ」というクイズ番組が編集部内でひそかなブームになっています．このクイズ番組，「ポップ・アート」や「ブランド」など，ジャンルを限定し，興味のない人にとっては問題の意味さえもわからないような，「狭くて・深い」問題ばかりが出題される，要するに「おたく」な番組です．これだけは人には負けたくないというものが，なにかひとつくらいは欲しいのですが，熱しやすくさめやすい性格ゆえ，カルトの道は遠そうです．(ま)

●休日にケーキを焼いた．1度めは書いてある手順の通りにやったのでうまく焼けた．調子にのって2度めは自家製マーマレードを加えるというアレンジを試みたところ，見事膨らまないままに終わった．料理の本というのは，写真入りで手順が記されていて，その通りにやればまず失敗しないのだけれど，自分なりに変更するには理屈を知る必要があるということなのだろう．でも，マズいCONFIG.SYSはリセットすればすぐなおせるけれど，料理にやりなおしはきかない……とほほ．(kos)

●もうすぐ冬が終る．でもアッと言う間に夏が来るだろう．夏が終れば秋，そして再び冬がやってくる．その頃には納車されるだろうと，一人目を細めて微笑む．「何，ディスプレイに向ってニヤけているの」の声で我に返る現実逃避の一瞬でした(笑)．(yoshi)

愛読者カードご記入のお願い

本誌をお買い上げいただき、誠にありがとうございます。
お手数ですが、今後の誌面づくりの参考にさせていただきたいと存じますので、
各欄にご記入の上、ご返送くださるようお願い申しあげます。

月刊プログラマーズ・ページ READER'S SURVERY CARD 愛読者カード

本誌の号数　　年　　月号

本誌を知った理由
- □1. 書店店頭　□2.他社出版物（　　　）
- □3.小社出版物　□4. DM　□5.口コミ

パソコンの所有形態（機種名）
- □1. 個人所有　（　　　）
- □2. 職場にある（　　　）
- □3. 持っていない

本誌を購入した書店　　市（区）　　書店

よく使用するソフトウェア

よく使用するOS

今月号でおもしろかった記事（いくつでも）

今後取り上げてもらいたい特集のテーマ

貴方が選ぶベスト３

パソコン雑誌	①	②	③
ソフトウェア	①	②	③
パソコン本体	①	②	③

本誌についてのご感想、お気づきの点、その他どのようなことでも結構ですので、ご意見をお聞かせください（「ドラゴン・ポスト＝読者の声」欄に掲載させていただく場合もあります。ペンネームをご希望の方はあわせてご記入ください）。

月刊プログラマーズ・ページ 広告資料請求カード

8	会社（学校）名		9	所属（学部）		10	従業員数	

※下記の該当する番号をご記入ください。

11	業種		12	職種		13	パソコン利用状況	

業種　①製造業　②金融・保険・不動産業　③商社・卸・小売業　④建設業　⑤通信・運輸　⑥システム・ソフト　⑦その他サービス業　⑧公務員　⑨その他

職種　①コンピュータ（ハード）技術者　②コンピュータ（ソフト）技術者　③エレクトロニクス技術者　④プログラマ　⑤その他の技術者　⑥専門職　⑦事務・営業職　⑧経営管理職　⑨商工サービス　⑩教職　⑪学生　⑫その他

パソコン利用状況　①会社で利用　②個人で利用　③会社でも個人でも利用　④利用していない

●希望される広告資料　　年　　月号

広告資料請求No. / 希望項目												
A 購入希望	A	A	A	A	A	A	A	A	A	A	A	A
B 説明希望	B	B	B	B	B	B	B	B	B	B	B	B
C 資料希望	C	C	C	C	C	C	C	C	C	C	C	C

定期購読のご案内

綴じ込みの振替用紙を使って、本誌の定期購読をお申し込みになれます。
1年間12冊＝11,760円（本体11,417円/税343円）
郵送料サービス

郵便はがき

料金受取人払
麹町局承認
3908
差出有効期間
平成4年11月
30日まで
切手不要

1 0 2 - 0 0

東京都千代田区平河町2-4-14
平河町KSビル
株式会社 翔泳社
マーケティング部 行

月刊プログラマーズ・ページ
READER'S SURVEY CARD　愛読者カード

氏名	性別 □1男 □2女	生年月日 19 年 月 日
住所 〒		電話 （　　）　－

職業			ユーザー区分
□1ソフト技術者	□2ハード技術者	□3技術系会社員	□1 プロ
□4事務系会社員	□5営業系会社員	□6教師	□2 マニア
□7一般公務員	□8経営管理職	□9理系学生	□3 一般ユーザー
□10文系学生	□11その他（　　）		

プレゼント記入欄	クイズ・ランダムスクリーン回答欄
希望商品番号	答

郵便はがき

料金受取人払
麹町局承認
3910
差出有効期間
平成4年11月
30日まで
切手不要

1 0 2 - 0 0

東京都千代田区平河町2-4-14
平河町KSビル
株式会社 翔泳社
月刊プログラマーズ・ページ
広告資料請求係行

＊当社で記入

1	SE	2	受付No.	

月刊プログラマーズ・ページ広告資料請求カード

3	会員番号	

＊すでに会員番号をお持ちの方はご記入ください。

4	フリガナ 氏名	姓　　名	5	生年月日	19 年 月 日

6	自宅住所	〒　電話（　　）
7	会社住所	〒　電話（　　）

資料送付希望先の番号に○印をつけてください。

払込票

通常払込料金加入者負担

口座番号 東京 4-15493

加入者名 株式会社 翔泳社

金額 11760

※

払込人住所氏名

受付局日附印

備考

切り取らないで郵便局にお出しください。

記載事項を訂正した場合は、その箇所に訂正印を押してください。

払込通知票

通常払込料金加入者負担

口座番号 東京 4-15493

加入者名 株式会社 翔泳社

金額 11760

料金 特殊 円

払込み

備考

受付局日附印

※（郵便番号 ）

TEL（ ）

払込人住所氏名

この払込通知票は、機械で使用しますので、下部の欄を汚さないよう特に御注意ください。また、本票を折り曲げたりしないでください。（郵政省）

各票の※印欄は、払込人において記載してください。

デジタル
会社

の名の示すとおり、大手商
より設立された会社だ。自
コン「VAR-MATE」シ
SER MEGA」シリー
ラー関係のシステムを開発
開発力を併せ持った独自の

泉田徹氏、システム事業部
展開と、今後の技術開発の
かった。

入の当初から関わって、導入の
いをしたり、またユーザーサポ
でも信頼されるような体制を敷
ます。Dell社は米国でもサポー
のしっかりしたメーカーとして
ているんですが、日本の場合に
一層、サポート重視でいかない
だと思っていますので。」と泉氏
。

ハードウェア的な付加価値を持つ商品を

、技術開発関係では、東京の技
ほか、大阪と福岡に技術部分室
し、システム開発に取り組んで

からの同社の事業展開についてうかがってみることにしよう。

「ちょうどAXが登場した時期に設立されたこともあり、パソコンはAX、プリンタのコントローラボードは独自に開発するということで、そのための開発環境を整えながら、当社のスタートをきったわけです。

その後、自社ブランドの20万円台の低価格ページプリンタMEGA Jr.などを発売し、シリーズ化を整え、また特定のシステム向けに専用のプリンタ

登富ということで、米国ではユーザーの好感度ナンバーワンに選ばれたりと、評価の高いAT互換機。同社はこのパソコンの総発売元となっている。

「Dell社のパソコンは、ハイエンドの機種でCPUに486を搭載し、50MHzのスピードを持っています。この高性能を生かし切れるシステムをハード・ソフト含めてお客さまに提供していきたいというのが私どもの考え方です。ですから、単に販社として動くというのではなく、お客さまに対してはシス

いる。現在、大きな柱となっているのは、プリンタコントローラ関係と、ボーリング場のオートスコアラーシステム関係の2つだ。システム事業部長の鶴田氏に、同社の開発関係の特色をあげていただこう。

「当社の技術部では今のところ、ロングタームなものへの取り組みが多いというのが特徴のひとつです。2年越しの仕事など、長期にわたる開発が中心ですね。

ボーリング場のオートスコアラーシ

定期購読のご案内

■定期購読は、1年間12冊と限定させていただきます。

■購読料のお支払は、この振替用紙をご利用ください。銀行振込、現金書留によるご送金は、ご遠慮ください。

■振替用紙に、ご住所、お名前をご記入のうえ、郵便局の貯金窓口にてお支払ください。

■小社より領収書の発行はいたしませんので、郵便局にて「郵便振替払込金受領証」をお受取ください。

■お申し込みいただいた方には、弊社より直接発送いたしますので、到着が発売日より遅れることがございますが、ご了承ください。

月刊プログラマーズ・ページ　定期購読直接申し込み書

この欄は、加入者あての通信にお使いください。

Programmer's Page MONTHLY MAGAZINE FOR PROGRAMMERS

年　　月号より

年間購読料 11,760 円（税・送料込み）

■ご希望の購読開始月号をご記入ください。

■送付先（住所・氏名は正確にご記入ください）

住所	□□□-□□	フリガナ	
		氏名	
		TEL	

この払込通知票は、機械で使用しますので、下部の欄を汚さないよう特に御注意ください。また、本票を折り曲げたりしないでください。　（郵　政　省）

郵便はがき

102-00

東京都千代田区平河町2-4-14
平河町KSビル

株式会社 翔泳社

月刊プログラマーズ・ページ
広告資料請求係行

2	受付No.	

…マーズ・ページ広告資料請求カード

＊すでに会員番号をお持ちの方はご記入ください。

名	5	生年月日	19　年　月　日

□10文系学生　□11その他（　　　　）　ユーザー

プレゼント記入欄	クイズ・ランダムスクリーン回答欄
希望商品番号	答

6	自宅住所	〒　電話（　　）	資料送付希望先の番号に○印をつけてください。
7	会社住所	〒　電話（　　）	

綴じ込みの

ちょっと気になるこの会社

三井物産デジタル株式会社

三井物産デジタル（株）は、その名の示すとおり、大手商社・三井物産(株)の100%出資により設立された会社だ。自社ブランド商品としてAXパソコン「VAR-MATE」シリーズやレーザープリンタ「LASER MEGA」シリーズを持つほか、スポーツ・レジャー関係のシステムを開発するなど、商社の営業力と技術開発力を併せ持った独自の展開に特徴がある。
東京・神田の本社に営業部長の泉田徹氏、システム事業部長鶴田進氏を訪ね、同社の事業展開と、今後の技術開発の方向性などについてお話をうかがった。

商社プラス技術開発力で新分野を開拓

三井物産デジタルは昭和63年に設立された若い会社だが、エレクトロニクス分野への取り組みは、同社設立の母体となった三井物産物資部の時代にさかのぼる。物資部という部署では、ICやIBM PC/AT互換機、周辺機器の販売、あるいは中国語のワープロ専用機を開発して中国へ輸出するなどのビジネスを行ってきたという経緯があり、この部門が独立するかたちで、三井物産デジタルが設立された。

まずは、営業部長の泉田氏に、設立からの同社の事業展開についてうかがってみることにしよう。

「ちょうどAXが登場した時期に設立されたこともあり、パソコンはAX、プリンタのコントローラボードは独自に開発するということで、そのための開発環境を整えながら、当社のスタートをきったわけです。

その後、自社ブランドの20万円台の低価格ページプリンタMEGA Jr.などを発売し、シリーズ化を整え、また特定のシステム向けに専用のプリンタを作らせていただいたり、コントローラボードだけをエンジンメーカー向けに作らせていただいたりと、そういった技術開発の力も高めてきました。

その一方で、スポーツ関連分野のシミュレーションのシステムの開発も行っています。三井物産の子会社に日本ブランズウィック（株）がありますが、ここのボーリング場のオートスコアラーシステムの開発も私どもが担当しています。」

こうした流れのなかで、最近力を入れているのが、米国Dell社のAT互換機の販売だ。Dell社のパソコンは、CPUも速く、バリエーションも非常に豊富ということで、米国ではユーザーの好感度ナンバーワンに選ばれたりと、評価の高いAT互換機。同社はこのパソコンの総発売元となっている。

「Dell社のパソコンは、ハイエンドの機種でCPUに486を搭載し、50MHzのスピードを持っています。この高性能を生かし切れるシステムをハード・ソフト含めてお客さまに提供していきたいというのが私どもの考え方です。ですから、単に販社として動くというのではなく、お客さまに対してはシステム導入の当初から関わって、導入のお手伝いをしたり、またユーザーサポート面でも信頼されるような体制を敷いています。Dell社は米国でもサポート体制のしっかりしたメーカーとして知られているんですが、日本の場合にはより一層、サポート重視でいかないとダメだと思っていますので。」と泉氏は語る。

ハードウェア的な付加価値を持つ商品を

一方、技術開発関係では、東京の技術部のほか、大阪と福岡に技術部分室を設置し、システム開発に取り組んでいる。現在、大きな柱となっているのは、プリンタコントローラ関係と、ボーリング場のオートスコアラーシステム関係の2つだ。システム事業部長の鶴田氏に、同社の開発関係の特色をあげていただこう。

「当社の技術部では今のところ、ロングタームなものへの取り組みが多いというのが特徴のひとつです。2年越しの仕事など、長期にわたる開発が中心ですね。

ボーリング場のオートスコアラーシ

ステムにしても、ハードの開発からすべてを行っていますので、短期間で仕上げられるものではないんです。また、こうしたシステムに使われている技術の中身を見ても、LANやデータベースも使っていますし、画像技術なども取り込んでいます。いわばコンピュータのエッセンスが集まっているようなものです。

こういう長期的なシステムの開発に取り組むメリットとしては、開発を通じて様々な新しい技術の蓄積が内部的に積み上げられつつある点ですね。開発力、技術力ということでは、自信を持てるようになってきました。

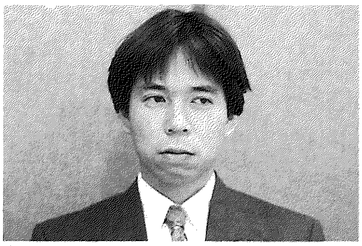
もうひとつ、当社の特徴として伸ばしていきたいのは、単にソフトウェアを開発するだけでなく、ハードウェア的な付加価値を持った商品を開発していくことです。たとえば、先ほどDell社のパソコンの話がでましたが、これまで286、386クラスのものでは、その能力からいってAXでも98でも、できることにあまり差が出てこなかったですね。それがDell社のAT互換機のように、486で50MHzという高性能になってきますと、そのハードウェアを生かすシステマティックな商品が作れる環境が生まれてきます。そこを狙った商品開発ができれば、当社の技術力と営業力がうまくタイアップできると期待しているんです。

スポーツ市場向けの新商品も開発

こうしたボーリング場向けシステムの開発などで培われた技術開発力をベースとして、同社ではオリジナル商品を出していく方向で取り組みが進んでいる。その第一弾が、ゴルフのスウィングチェッカーシステム「MOTION TRACER」だ。ゴルフ練習場などでゴルフスウィングを録画・再生のするこのシステムは、従来のビデオ方式に代わってデジタル方式を採用している。CCDカメラで捉えた映像をコンピュータのメモリに一時記憶し、必要な場面を呼び出して9コマのストロボ画面に瞬時に表示するシステムだ。ビデオ方式に比べて耐久性、信頼性に優れ、頻繁な再生にも応える操作性のよさを備えている。

この商品は今年の2月に発売されたばかりだが、テレビで紹介されたり、写真週刊誌に取り上げられたりと話題を呼び、ゴルフ練習場経営者などユーザーにも反応は上々のようだ。

このような新商品の開発について、泉田氏はこう語る。



営業部長 泉田徹氏

「当社としては、いわゆる開発会社というよりは、自社のオリジナル商品の販売をメインにしていきたいといった考えがありますから、今後も低価格のプリンターであるとか、高性能なパソコンであるとか、付加価値のついたシステムであるとか、そういうものを自社で作って、販売していくことを主体として展開したいと思っています。

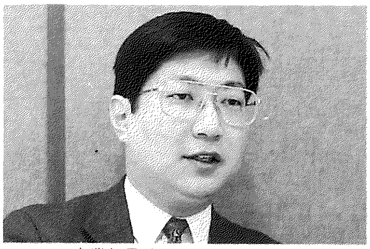
それが、商社のノウハウと技術開発力がうまく結びついた形で進んでいけば、おもしろいと思うんですよ。」

今後が楽しみなオリジナル商品開発

オリジナル商品の開発については、技術開発を担当する鶴田氏も積極的だ。

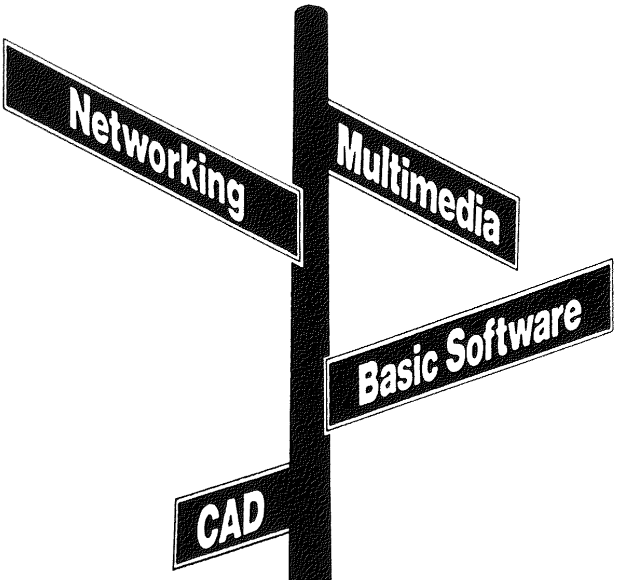
「今年から来年にかけて考えているのは、ロングタームでシステム開発をやりながら、その技術開発のなかから落とし子をいくつ生み出せるかということです。それが多いほど結果的に商品として売るものも多くなってくるわけですね。またその辺が一番難しいところでもあり、われわれも日々頭を悩ましているわけですが。

その中で、具体的にはDell社の高性能パソコンとどのようにリンケージさせていけば、より付加価値の高いシステムができるか、これは是非形にしてみたいですね。単にスピードが速いパソコンを売るというだけでは、今後やっていけないと思うんですよ。そういう高速なパソコンで何ができるのか、それをやりたいのなら当社から買うしかないよ、と言えるだけの付加価値のついた魅力ある商品を育てたいと考えています。」



システム事業部長 鶴田進氏

商社のノウハウと技術開発力とを合体させた分野を切り開きたいと、意気盛んな両氏。さて、次はどんな商品が飛び出してくるのか、たいへん楽しみなところである。



高水準の技術開発にはパーフェクト精神が根幹です。

ワンランク上の提案型ソフトウェアハウス＝エルテックスは、ユーザーや上司の求めている以上の開発を追求していく、我がままプロ集団の組織。常に高レベルの開発を行っていくためには、たえず外部の開発にアンテナを張りめぐらしておかなくてはなりません。その一環として、毎年数名、ラスベガスのコムデックスショー等、1週間程度の視察旅行に参加します。パーフェクトなものを求め続けていく技術者たちに、私たちは、たえず評価を与えていきます。

技術者随時募集

会社概要

ELTEX

設　立	昭和60年12月14日
資本金	1000万円 １ネットワークの端末システムの開発 ２ＣＤ－ＲＯＭ、ＬＤなど新しいメディアに対応したシステムの開発 ３ＯＣＤ、ＣＡＤなどのソフトウェアプロダクトの開発、販売 ４ビジネスパッケージの開発、販売
特　徴	大手クライアント向けにネットワークやマルチメディアなどのシステム開発を幅広く対応している。また、手書文字読み取りシステム「読造」*1、水道工事店向け「配管ＣＡＤ」*1などの自社ソフトウェア製品の開発販売や、電子回路ＣＡＤ「Ｔａｎｇｏ」シリーズ*2の日本語化など積極的な事業展開をしている。 *1　当社で開発し三菱系ディラーで販売しているプロダクト *2　米アクセルテクノロジ社の製品をＰＣ9801への移植と日本語化したプロダクト、㈱ソーテック工人舎事業部で販売している。

募集要項

職　種	①システムエンジニア　②プログラマー
資　格	専門卒以上20～36歳の方　※Ｃ言語・アセンブラ言語経験者大歓迎　※未経験者でも好奇心・向学心の旺盛な方は歓迎します。
給　与	17万円（専門卒初任給）～46万円（36歳課長、諸手当含む） 24歳例：23万5000円（情報処理1種手当・住宅手当含む） 30歳例：31万6000円（中級SE、情報処理1種・住宅手当含む）
手　当	時間外手当、役職手当（5000円～６万円）、住宅手当（１万5000円）、資格手当（２種：5000～特種：２万円）、通勤手当
賞　与	賞与年２回（平成１年度実績年間５ヵ月）、決算賞与年１回（平成１年度実績１ヵ月）
勤務地	本社（横浜駅近く） ◇派遣・出向はありません。
勤務時間	フレックスタイム制（コアタイム10:00～16:00）試用期間中は標準時間9:00～17:45
休日休暇	完全週休２日制（土・日）、祝日、年３回連休（GW・夏期・年末年始）、有給・特別・休暇
福利厚生	各種社会保険完備、退職金制度、財形貯蓄、社員旅行、海外研修、定期健康診断、社員寮、スポーツクラブ法人会員
応募方法	電話連絡の上、履歴書（写真貼付）、職務経歴書を持参し来社ください。郵送も受け付けます。
連絡先	〒220 横浜市西区南幸二丁目20番５号　東伸24ビル５階　℡(045)322-9667　総務課

採用HOT・LINE

TEL.045-322-9667

株式会社エルテックス

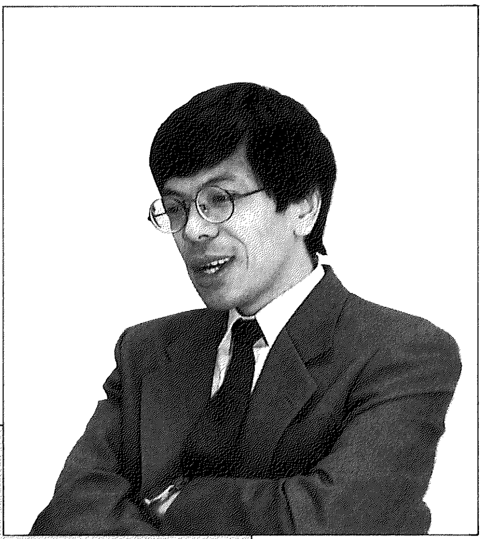
Microsoft®

マイクロソフトの実現

手書き文字認識、アプリケーションインテグレーション、
オブジェクト指向テクノロジー、次世代GUI、
そしてマルチメディア。
あなたは何を実現しますか?

募集要項

ネットワーク開発エンジニア	●次のいずれかに2年程度の経験がある方 (ネットワークに興味がある方歓迎) -NOS -トランスポーテーションスタック -LU6.2 -データベース -SQL言語
OS開発エンジニア	●次のいずれかに2年程度の経験がある方 (マルチタスクOSの開発に興味がある方歓迎) -UNIX移植 -BIOS -グラフィックドライバー -アウトラインフォント -マルチCPU -WYSIWYG ●C、C++、x86アセンブラのプログラミング経験がある方 ●ウィンドウシステム(Windows等)の開発の経験または知識がある方
システム製品プログラムマネージャ	●システムソフトウェアの開発マネージメントの経験がある方 ●C言語、x86アセンブラ言語の経験がある方 ●PC、ワークステーションのハードウェアに関する知識がある方 (英会話に自信のある方歓迎)
システム製品テスティングエンジニア	●マルチタスクOS(ウィンドウシステムを含む)、ネットワークのテスティング経験がある方 (興味がある方歓迎) ●C言語のプログラミング経験または知識がある方 (英語のリーディング、ライティングができる方歓迎)
アプリケーション製品/言語製品開発エンジニア及びプログラムマネージャ	●C、8086系アセンブラのプログラミングの経験がある方 ●C++を使ったプログラミングの経験がある方 ●Windows上でのプログラミングの経験がある方 ●ソフトウェアの設計仕様作成の経験がある方 (英語のリーディング、ライティングができる方歓迎)
アプリケーション製品/言語製品プロダクトマネージャ	●パソコンソフトウェアのプロジェクトリーダー(製品主幹)の経験がある方 ●パソコンソフトウェアの企画の経験がある方
アプリケーション製品テスティングエンジニア	●アプリケーションソフトのテスティング経験がある方 (興味がある方歓迎) ●DOS/Windows環境の技術知識がある方 ●マクロ言語を使用した実務プログラミング経験または知識がある方
データベース管理ソフト開発エンジニア	●スプレッドシート、ワープロの開発の経験または知識がある方 ●データベースソフトの開発の知識がある方 ●データベースソフトの理論について勉強したことがある方
テクニカルライター	●ソフトウェアのマニュアルの企画、編集、製作、原稿のライティングなどの経験がある方
テクニカルサポートスタッフ	●パソコンの基礎知識がある方
製品教育課スタッフ	●アプリケーションソフトのインストラクタ経験がある方



第二研究開発部
システム1課　課長
柿園　昭俊
(1987年入社)

「現在、システム製品の開発に携わっています。具体的にはMS-DOSとWindows for Penの開発ですが、多くのユーザーに対して影響力を持つ製品ですので、責任感のある仕事であると同時に魅力的な仕事ですね。

MS-DOSについて言えば、常に増え続けるエンドユーザーのニーズをプロダクトに反映させることが大事ですね。理想としては、システム製品としての存在を意識させないぐらい完成度を高めた商品にすることです。たとえば今回のMS-DOS 5.0のMS-DOSシェルなどは一つの試みと言えますね。また、Windows for Penは、コンピュータの入力手段の1つとして、ペンを使い直接文字を書くアプローチを採ったものです。手で文字を書くという人間の自然の行為とコンピュータのスムーズなインターフェースを採ることが、このシステムのポイントです。

マイクロソフト社ではいままで経験しなかった新しい文化、新しいテクノロジーに触れることができます。このような環境であなたの経験を生かしてみませんか。　あなたがマイクロソフト社で何かを実現するということは、コンピュータ業界において影響力の強いプロダクトを実現するということなのです。」

募集要項

職種■図表を参照ください。
資格■22歳から30歳まで
くわしくは図表を参照ください。
給与■経験・能力を考慮の上、当社規定により優遇します。
固定給制　23万2,000円以上('91年実績)
待遇■昇給年1回、賞与年2回、各種社会保険完備、交通費全額支給、各種制度:退職金制度、財形貯蓄、社員持株制度、社内旅行、スポーツクラブ会員、団体定期保険、医療保障保険有(会社負担)
勤務地■本社(新宿)
勤務時間■フレックスタイム制
標準勤務時間/9:00～17:30
*コアタイム11:00～15:00(標準労働時間7.5h)
休日■完全週休2日制(土・日)、祝日
休暇■年末年始、夏季、慶弔、特別、年次、有給休暇、傷病休暇(有給)
応募■希望職種明記の上、履歴書(写真貼付)と職務経歴書をご郵送ください。
*追って面接日等、連絡いたします。
*入社日相談に応じます。
*応募の秘密、厳守します。
*応募の書類は返却しません。
*お問い合わせ、歓迎します。お気軽にお電話ください。
*特に必要な場合、連絡方法を明記してください。
交通■新宿オフィス:JR新宿駅より徒歩10分、新大久保駅より徒歩3分、西武新宿駅より徒歩3分
芝オフィス:JR浜松町駅より徒歩10分、都営三田線芝公園駅より徒歩10分、浅草線大門駅徒歩10分

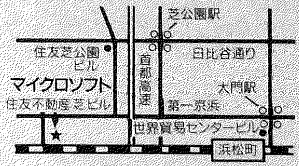
会社概要

設立■1986年2月17日
資本金■1億円
売上高■46億円('89年実施)
66億円('90年実施)
90億円('91年6月期実施)
従業員数■279名
平均年齢■29歳
米国本社■マイクロソフト・コーポレーション
(ワシントン州レッドモント)
事業内容■パソコン用ソフトウェアの開発・販売
(MS-DOSをはじめWindows、アプリケーションではExcel、Word、Multiplan、Works、言語ではCやBasicなど世界標準となる製品を提供)

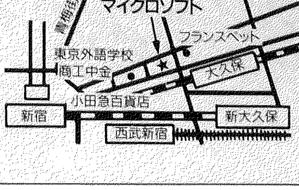
採用関係連絡先

〒105 東京都港区芝1-11-11
住友不動産芝ビル
Tel.03-5484-1420　人材開発部S係
仕事内容などのご質問もお気軽にどうぞ。

芝オフィス
〒105 港区芝1-11-11 住友不動産芝ビル
Tel.03-5484-1420(人材開発部)

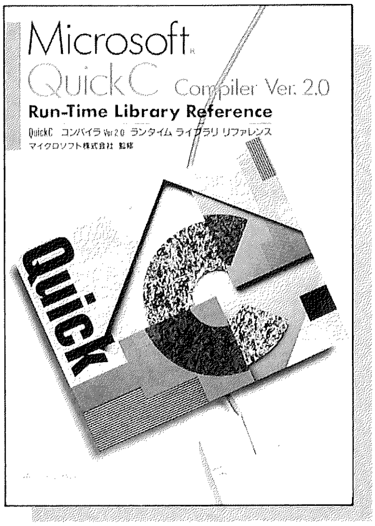


新宿オフィス(勤務地)
〒160 新宿区西新宿7-5-25 Kビルディング
Tel. 03-3363-1200(代表)



プログラミングのための優良図書

SE
Shoeisha Co., Ltd.



Quick C コンパイラ Ver.2.0 ランタイムライブラリリファレンス

Quick C Compiler Ver.2.0
Run-Time Library Reference

マイクロソフト株式会社　監修
A5判/364頁
定価2,400円

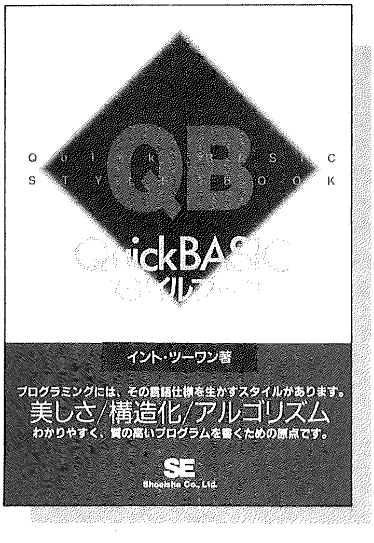
『言語ソフトのベストセラー「Microsoft Quick C」のオンラインヘルプを完全書籍化。マイクロソフト社監修によるパーフェクトQuick C リファレンス。プログラミング時の参考書として、また学習書として活用できます。』

Quick BASIC Ver.4.5 ステートメント&関数リファレンス

Quick BASIC Ver.4.5
Statement & Function Reference

マイクロソフト株式会社　監修
A5判/394頁
定価2,400円

『Quick BASIC Ver.4.5のQBアドバイザを完全書籍化。パソコンの前を離れて参照したい時など学習用として最適です。各ステートメント & 関数にサンプルプログラミングを満載したパーフェクトリファレンス。マイクロソフト社監修。』



Quick BASIC スタイルブック

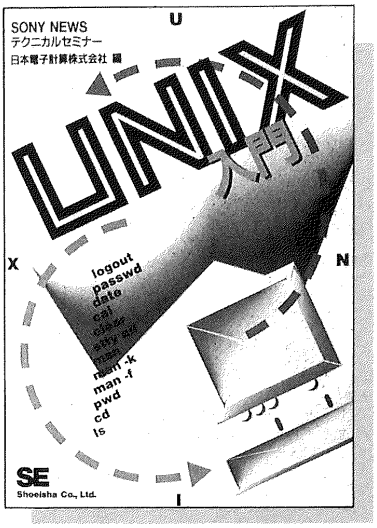
イント・ツーワン　著
B5判変型/236頁
定価2,300円

『初心者からマニアまで幅広く使われるQuick BASIC。このQuick BASICを用いて質の高いプログラミングを書くためのノウハウがいっぱいつまっています。一問一答の問題集形式。パズルのようにプログラミングが学べます。』

UNIX入門
―SONY NEWSテクニカルセミナー―

日本電子計算株式会社　編
A5判/237頁
定価1,800円

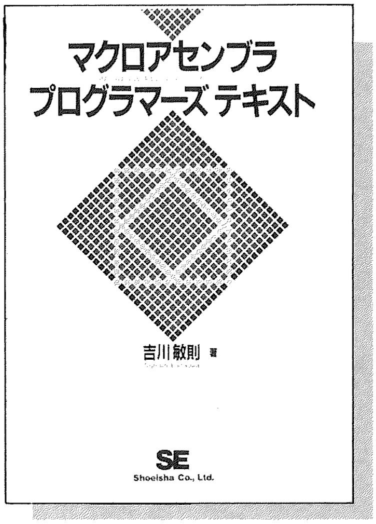
『ソニーのNEWSによる「UNIX入門」セミナーのテキストをベースとした入門書。UNIXのコマンド、エディタの使い方、日本語入力の方法、ネットワークの利用方法などを豊富な実例をまじえて解説。はじめてのワークステーションを利用するユーザ必携の書。』



マクロアセンブラ プログラマーズテキスト

吉川敏則　著
A5判/243頁
定価2,600円

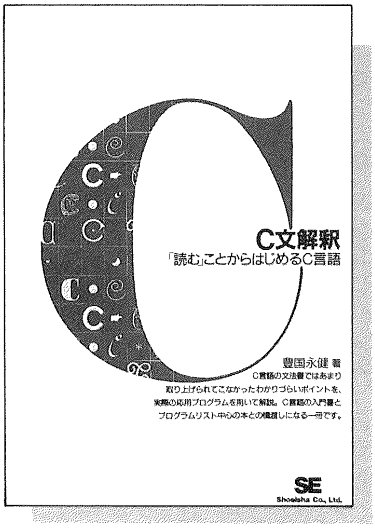
『最低限必要なハードとソフトの知識から、データ演算の仕組みを踏まえた上で、アセンブラプログラミングの基本を丁寧に解説。アセンブリ言語に興味を持つ人や、これからアセンブリ言語でプログラムを作成する人に役立つ一冊。』



C文解釈
―「読む」ことからはじめるC言語―

豊国永健　著
B5判/183頁
定価2,000円

『C言語の文法書ではあまり取り上げられてこなかったわかりづらいポイントを、実際の応用プログラムを用いて解説。C言語の入門書とプログラムリスト中心の本との橋渡しになります。』



SE Shoeisha Co., Ltd. 株式会社翔泳社 〒102東京都千代田区平河町2-4-14平河町KSビル TEL.03-3263-0447(代表)
翔泳社の本は全国どの書店でもお求めになれます。店頭にない場合は「至急」とお申し出の上ご注文いただければ10日位でお届けできます。

マルチメディア時代の先端を走る

SOFT VISION

プログラマーズ・ページ読者 全国一斉ソフトビジョン入社テスト

① 昨年日本が米国に払ったソフトのライセンス料は何億ドル?

② ソ連にプログラマーは何人いますか?

③ マニュピレーターってなにする人?

④ おなじみプログラマーズページの表紙モデルは誰?

⑤ ソフトビジョンの"創立者"はどの人でしょう。

この問題のカルト度は㊥です。一問、二問はあたり前!!

ソフトビジョンの求人は、「応募者が自らプレゼンテーションを行う」超前向きな採用を推進しています。上記の問題、3問以上解けた方だけが、当社に対して自分自身をプレゼンテーションする権利を獲得することができるのです。3問解けたら電話連絡の上、面接にはりきってどうぞ。

はっきり言って、これは理不尽な求人広告です。

会社プロフィール

設立●昭和59年4月2日 代表者●代表取締役 鈴木和芳 資本金●990万円 従業員数●144名(男:105名、女:39名) 平均年齢●25.5歳 売上高●8億8000万円(平成3年度実績) 事業内容●①CDIソフトの開発 ②FA&ロボットなどの自動制御 ③データ通信&自動車電話などの通信制御 ④TV局向け&医療向けの映像制御・画像処理 ⑤オンライン端末&無線端末などのマイクロソフト制御 上記分野を中心としたソフトウェア・ハードウェアの企画、開発、販売、コンサルティング 主要機種●HP64000、HP9000LAN、PC9801、FMR、Macintosh他多数 主要言語●C、アセンブラ、他 主要取引先●日本電気㈱、日本電気無線電子㈱、日本無線㈱、富士通機電㈱、松下システムエンジニアリング㈱、松下通信工業㈱、京セラ㈱、JRCエンジニアリング㈱、東芝エンジニアリング㈱、昭和電子㈱

募集要項

職種●コンピュータエンジニア ※今春・来春新卒者のお問い合せ歓迎 ※学生アルバイト同時募集 資格●学歴不問 26歳位迄 給与●月給 固定給制19万円以上 ※経験・年齢・前職を考慮の上、相談により決定します。26歳支給例/31万6500円 アルバイト/時給1000円以上(時間相談に応じます。) 待遇●昇給年1回(3年度実績12%)、賞与年2回(2年度実績7ヵ月)、通勤交通費(2万4000円迄支給)、各種社会保険完備、住宅・家族・精勤・時間外・役職手当、退職金制度、財形貯蓄制度、社内旅行(グアム)、クラブ活動(野球部、テニス部、バスケット部、ウインドサーフィン部 他多数クラブ ※会社から援助金支給)、ラップトップ・レンタル制度(長期間無料) 勤務地●本社(相模原) 勤務時間●フレックスタイム制(コアタイム10:30〜15:30) ※タイムカードはありません。1日標準労働時間8時間 休日休暇●完全週休2日制(土・日)、祝日、夏期・年末年始・GW・有給(初年度8日から)・慶弔休暇 ※年間125日+有給 応募●電話連絡の上、履歴書(写真貼付)・職務経歴書(職歴のない方は自己PR書)を持参ください。郵送も可。 ※FAX応募も歓迎します。 ※応募の秘密厳守します。 ※入社日は相談に応じます。 ※新卒者の資料請求歓迎します。 ※大学生の会社訪問歓迎します。 交通●JR横浜線相模原駅徒歩5分

採用関係連絡先

〒229 神奈川県相模原市相模原4-3-14相模原第一生命ビル「株式会社ソフトビジョン 人事部人事課」 フリーダイヤル0120-211744 採用担当/海老沢、斉藤

FAX 0427-51-3333

株式会社 ソフトビジョン

Microsoft Macro Assembler Professional Development System Version 6.0 マクロ/ディレクティブ/演算子一覧

MS-DOSマクロ

●コンソール入出力

マクロ	説明
@GetChar	キーボードから文字を取得
@GetStr	キーボードから文字列を取得
@ShowChar	画面にいくつかの文字を表示
@PrtChar	プリンタにいくつかの文字を出力
@ShowStr	終端が$の文字列を表示

●デバイスとファイルの制御

マクロ	説明
@Read	ファイルやデバイスからデータを読み出す
@Write	ファイルやデバイスにデータを書き込む
@MakeFile	ファイルを作成
@OpenFile	入出力ファイルをオープン
@CloseFile	オープンしているファイルハンドルをクローズ
@DelFile	指定されたファイルを削除
@MoveFile	ファイルの移動, ファイル名の変更
@GetFirst	最初に一致したファイルを取得
@GetNext	次に一致したファイルの情報(ディレクトリエントリ)を取得
@GetDTA	ディスク転送アドレス(DTA)を取得
@SetDTA	ディスク転送アドレス(DTA)を設定
@GetFileSize	ファイルの大きさを取得
@MovePtrAbs	ファイルポインタを絶対位置により移動
@MovePtrRel	ファイルポインタを相対位置により移動

●ディレクトリとドライブの制御

マクロ	説明
@MkDir	ディレクトリを作成
@RmDir	ディレクトリを削除
@ChDir	カレントディレクトリを変更
@GetDir	指定されたドライブのカレントディレクトリを取得
@GetDrv	カレントドライブを取得
@SetDrv	カレントドライブを設定
@ChkDrv	ディスクに関する様々な情報を取得

●メモリ制御

マクロ	説明
@FreeBlock	メモリブロックを解放
@GetBlock	メモリブロックを割り当てる
@ModBlock	割り当てられたメモリブロックを変更

●そのほかのMS-DOSマクロ

マクロ	説明
@GetDate	システムの日付を取得
@SetDate	システムの日付を設定
@GetTime	システムの時刻を取得
@SetTime	システムの時刻を設定
@GetVer	MS-DOSのバージョンを取得
@GetInt	指定された割り込みのベクタを取得
@SetInt	指定された割り込みのベクタを設定
@Exit	終了コードを返してMS-DOSに戻る
@TSR	プログラムをメモリに常駐させたまま終了

アセンブリ言語の定義済みシンボル

●セグメント情報のための定義済みシンボル

シンボル	説明
@code	.CODEセグメントのセグメント名
@CodeSize	コードセグメントのサイズ
@CurSeg	現在のセグメントのセグメント名
@data	DGROUPセグメント(.DATA, .DATA?, .CONST)のセグメント名
@DataSize	DGROUP(.DATA, .DATA?, .CONST)のセグメントサイズ
@fardata	.FARDATAセグメントのセグメント名
@fardata?	.FARDATA?セグメントのセグメント名
@Model	メモリモデル
@stack	スタックセグメントのセグメント名
@WordSize	現在のプロセッサのワードサイズ

●環境情報のための定義済みシンボル

シンボル	説明
@Cpu	CPUの型
@Environ	環境変数の値
@Interface	言語型およびOSの型
@Version	アセンブラのバージョン番号

●日付情報と時刻情報のための定義済みシンボル

シンボル	説明
@Date	システムの日付
@Time	システムの時刻

●ファイル情報のための定義済みシンボル

シンボル	説明
@FileCur	現在のファイルのファイル名
@FileName	現在のファイルのベース名
@Line	モジュール中の現在の行番号

●マクロ文字列操作のための定義済み関数

シンボル	説明
@CatStr	文字列の連結
@InStr	文字列中の別の文字列の位置
@SizeStr	文字列のサイズ
@SubStr	文字列から取り出した文字列

●そのほかのシンボル

シンボル	説明
$	ロケーションカウンタの現在の値
?	初期化されないデータを示す

ディレクティブ

●OPTIONディレクティブ

M510互換オプション

オプション	デフォルト
CASEMAP:maptype	NOTPUBLIC
DOTNAME¦NODOTNAME	NODOTNAME
M510¦NOM510	NOM510
OLDMACROS¦NOOLDMACROS	NOOLDMACROS
OLDSTRUCTS¦NOOLDSTRUCTS	NOOLDSTRUCTS
SCOPED¦NOSCOPED	SCOPED

プロシージャの使用に対するオプション

オプション	デフォルト
LANGUAGE:langtype	なし
EPILOGUE:macroid	OSおよびプロセッサの型に依存
PROLOGUE:macrofuncid	OSおよびプロセッサに依存
PROC:visibility	PUBLIC

そのほかのオプション

オプション	デフォルト
EXPR16¦EXPR32	プロセッサに依存
EMULATOR¦NOEMULATOR	NOEMULATOR
LJMP¦NOLJMP	LJMP
NOKEYWORD:<keywordlist>	なし
NOSIGNEXTEND	オフ
OFFSET:offsettype	GROUP
READONLY¦NOREADONLY	NOREADONLY
SEGMENT:segmentsize	USE16

●条件付きディレクティブ

条件付きアセンブルディレクティブ

ディレクティブ	説明
IF	式が真(0でない)のときブロックをアセンブル
IFE	式が偽(0)のときブロックをアセンブル
IFB	式が空のときブロックをアセンブル
IFNB	式が空でないときブロックをアセンブル
IFDEF	シンボルが定義済みのときブロックをアセンブル
IFNDEF	シンボルが未定義のときブロックをアセンブル
IFDIF	引数が異なるときブロックをアセンブル
IFIDN	引数が同一のときブロックをアセンブル
ELSE	IFステートメントが偽のときの条件ブロックを開始
ELSEIF	IFステートメントが偽のときの新しい条件を開始
ENDIF	条件ブロックを終了
	それぞれのIFには, ELSEが付いたものもあり

条件付きエラーディレクティブ

ディレクティブ	説明
.ERR	式が真(0でない)のときエラーを生成
.ERRE	式が偽(0)のときエラーを生成
.ERRB	式が空のときエラーを生成
.ERRNB	式が空でないときエラーを生成
.ERRDEF	シンボルが定義済みのときエラーを生成
.ERRNDEF	シンボルが未定義のときエラーを生成
.ERRDIF	引数が異なるときエラーを生成
.ERRIDN	引数が同一のときエラーを生成

●簡略化セグメント

ディレクティブ	説明
.MODEL	メモリモデル, 言語, OS, スタックを設定
.STARTUP	セグメントレジスタとスタックの初期化を設定
.EXIT	プログラムを正常終了
OPTION PROLOGUE	.STARTUPのプロローグマクロを定義
OPTION EPILOGUE	.STARTUPのエピローグマクロを定義
.CODE	CODE セグメントを開始
.STACK	STACK セグメントを開始
.DATA	初期化されるDATAセグメントを開始
.DATA?	初期化されないDATAセグメントを開始
.CONST	定数DATAセグメントを開始
.FARDATA	初期化されるfarデータセグメントを開始
.FARDATA?	初期化されないfarデータセグメントを開始
ASSUME	レジスタの仮定を宣言
ENDS	現在のセグメントを終了

●セグメント

ディレクティブ	説明
SEGMENT	セグメントを開始
ENDS	セグメントを終了
GROUP	複数のセグメントを結合
ASSUME	デフォルトのセグメントレジスタを設定
END	ソースファイルを終了
.DOSSEG/.ALPHA/.SEQ	セグメントの順序付けを設定

●複合データ型

ディレクティブ	説明
STRUCT	データ構造体の定義
UNION	共用体の定義
RECORD	データレコードの定義
TYPEDEF	データ型の定義
ENDS	データ構造体の定義の終了

MASK	指定したフィールドのビットに対応するビットマスクを返す
WIDTH	レコードまたはレコードフィールド幅を返す
SIZEOF	変数に割り当てられたバイト
LENGTHOF	構造体のデータ項目の数
ALIGN	コード／データを指定された境界に合わせて配置
EVEN	コード／データをワード境界で配置
ORG	ロケーションカウンタを設定

●プロシージャ／コードラベル

ディレクティブ	説明
PROC	プロシージャブロックを開始
ENDP	プロシージャブロックを終了
LOCAL	ローカルプロシージャのプレースホルダを宣言
PROTO	プロシージャのパラメータのプロトタイプを定義
INVOKE	スタック形式のプロシージャ呼び出し
LABEL	現在のセグメントでラベルを宣言
:/::	ラベルの区切り

●適用範囲と可視性の制御

ディレクティブ	説明
PUBLIC	パブリックシンボルを宣言
EXTERN[DEF]	外部シンボルを宣言
COMM	共有シンボルを宣言
INCLUDE	アセンブラフォーマットのファイルをインクルード
INCLUDELIB	ライブラリフォーマットのファイルをインクルード

●フロー制御ディレクティブ

ディレクティブ	説明
.IF	IF 条件コードを生成
.ELSE	ELSE 条件コードを生成
.ELSEIF	ELSEIF 条件コードを生成
.ENDIF	条件コードを終了
.WHILE	WHILE ループコードを生成
.ENDW	WHILE ループの終わりを表示
.REPEAT	REPEAT ループコードを生成
.UNTIL[CXZ]	REPEAT ループの終了条件をチェックするコードを生成
.BREAK	ループを脱出
.CONTINUE	ループ評価コードにスキップ

●マクロと繰り返しブロック

ディレクティブ	説明
MACRO	マクロブロックを開始
ENDM	マクロブロックを終了
PURGE	マクロをメモリから削除
TEXTEQU	テキストシンボルや式を処理しテキストシンボルを定義
LOCAL	マクロのローカルラベル／シンボルを宣言
PUSHCONTEXT	スタックにコンテキスト情報を保存
POPCONTEXT	スタックに保存されているコンテキスト情報を復元
EXITM	マクロを終了
GOTO	マクロ中でラベルにスキップ
REPEAT	指定された回数だけアセンブルを繰り返し
WHILE	式が真の間アセンブルを繰り返し
FOR	指定されたパラメータに対してマクロを繰り返し
FORC	指定された文字に対してマクロを繰り返し

●単純データ型

ディレクティブ	説明
(S)BYTE/(S)WORD/(S)DWORD/FWORD/QWORD/TBYTE/REAL4/REAL8/REAL10	1つまたは複数のグローバルデータを割り当て初期化。Sプレフィックス付きは符号付きデータを示す
DB/DD/DF/DQ/DT/DW	データ型の定義

●リスティングの制御

ディレクティブ	説明
TITLE/SUBTITLE	プログラムのリスティングタイトル／サブタイトルを設定
PAGE	ページ長とページ幅を設定
.LIST	ステートメントのリスティングを開始
.LISTALL	すべてのステートメントのリスティングを開始
.LISTMACRO	マクロのリスティングを開始
.LISTMACROALL	コメントを含めたマクロのリスティングを開始
.LISTIF	式が真のときマクロのリスティングを開始
.NOLIST	すべてのステートメントのリスティングを停止
.TFCOND	偽の条件ブロックのリスティングを切り換え
.CREF	シンボルテーブルおよびブラウザへのシンボルのリスティングを開始
.NOCREF	シンボルテーブルおよびブラウザへのシンボルのリスティングを停止

●プロセッサ

ディレクティブ	説明
.186/.286/.386/.486	各CPUの命令(特権命令は除く)をアセンブル可能にし，上位CPUの命令を禁止する
.287/.387/.8087	各FPUの命令をアセンブル可能にし，上位FPUの命令を禁止する
.286P/.386P/.486P	各CPUの全命令をアセンブル可能にし，上位CPUの命令を禁止する

●そのほかのディレクティブ

ディレクティブ	説明
EQU	シンボルを定義(数値シンボルは再定義不可)
=	シンボルを定義(数値シンボルは再定義可)
TEXTEQU	テキストマクロや式を処理しテキストマクロを定義
OPTION	アセンブラオプションを設定
CASEMAP	小文字を大文字に変換
FLOAT	浮動小数点演算命令の生成を制御
.RADIX	定数に対してデフォルトの基数を設定
CATSTR	文字列を連結
INSTR	文字列中の文字の位置
SIZESTR	文字列の長さ
SUBSTR	文字列から特定の文字を取り出し
COMMENT	コメントブロックを開始
ECHO	アセンブル中にメッセージを表示
PUSHCONTEXT	スタックにコンテキスト情報を保存
POPCONTEXT	スタックに保存されているコンテキスト情報を復元

演算子

●アドレス演算子

演算子	説明
[]	囲まれた式をインデックスとして扱う
:	セグメントをオーバーライド
OFFSET	式のオフセットアドレスを与える(アセンブラが計算)
LROFFSET	式のオフセットアドレスを与える(リンカが解決)
SEG	式のフレームを返す
THIS	現在のオフセットとセグメントを返す
SHORT	ショートラベル(次の命令の先頭からの距離が128バイト未満のとき)を定義
ADDR	INVOKE ディレクティブにラベルのアドレスをわたす

●型演算子とサイズ演算子

演算子	説明
SIZEOF	変数に割り当てられたバイト数
LENGTHOF	構造体のデータ項目の数
WIDTH	レコードまたはレコードフィールドの幅(ビット数)
TYPE	式の属性
NEAR	セグメントの内側(距離)
FAR	セグメントの外側(距離)

●マクロ演算子

演算子	説明
&	置き換えられたシンボルの結合
< >	引数<text>を単一のシンボルとして扱う
!	引数<text>をリテラル文字として扱う
%	引数<text>を式として扱う
;;	マクロコメントを開始(リスティングファイル中には展開しない)

●文字列演算子

演算子	説明
@CatStr	文字列を連結
@InStr	文字列中で文字の位置を探す
@SizeStr	文字列の長さを返す
@SubStr	文字列から文字を取り出す
'(シングルクォーテーション)/"(ダブルクォーテーション)	文字列を示す
!	引数<text>をリテラル文字として扱う

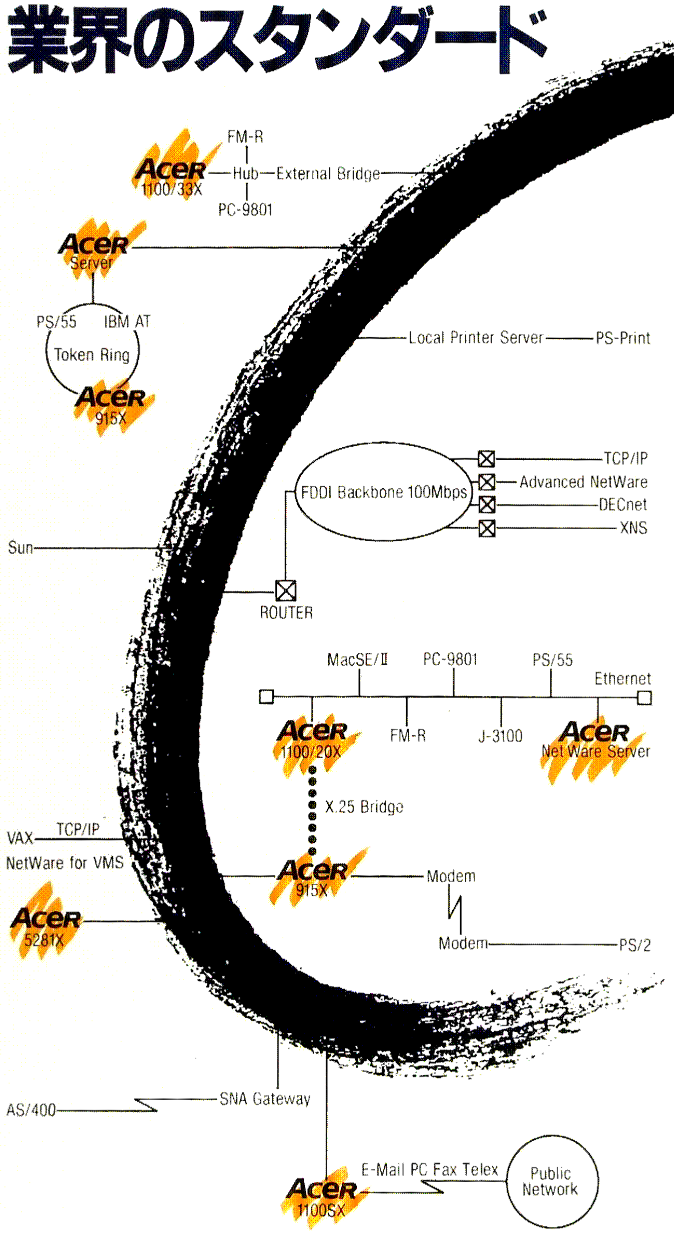
●そのほかの演算子

演算子	説明
HIGH/LOW	式の上位／下位バイトを返す
HIGHWORD/LOWWORD	式の上位／下位ワードを返す
PTR	式が指定された型を持つものとして扱う
MASK	指定したフィールドのビットに対応しているビットマスクを返す
WIDTH	レコードまたはレコードフィールド幅を返す
DUP	1つの値を繰り返し宣言
THIS	現在のオフセットとセグメントの値を持つオペランドを返す
OPATTR/.TYPE	式のモードと通用範囲を返す
SHORT	ショートラベル(次の命令の先頭からの距離が128バイト未満のとき)を定義
.	構造体のフィールドを区切る
?	初期化されないデータを示す
¥	ステートメントを次の行に続ける
$	ロケーションカウンタの現在の値を与える
;	コメントを開始
+/-/*//	加減乗除を行う

Net Ware

Acer
The Word for Value

パソコンLANにおける業界のスタンダード



THE ARCHITECTURE EXTENDED-AX

世界のパソコンLAN市場をリード

米国Novell社の"NetWare"は、世界のパソコンLAN市場において60%以上の圧倒的なシェアをもつLAN OSです。
このNetWareに日本語が付加されたことによってACERなどのIBM-PC(AT互換機)はもちろんのことAXマシン、PC-9801、FM-R、PS-55、J-3100などの国産PCへの対応も広がっており、日本語・英語両環境でのネットワーク構築を可能といたしました。

NetWareの特徴

NetWareは、ミニコンやメインフレームでしか提供できなかったさまざまな機能をパソコンベースで実現することができます。他のLANシステムと比較した場合の大きな特徴の一つにシステム・フォルト・トレラント(SFT)機能があげられます。例えばファイルサーバーに2台のハードディスクを装着させることによって"ディスクの二重化"が可能であり、常に同じ情報が2台のハードディスクに格納されるため、万一メインのハードディスクに故障が生じても、サブ側のハードディスクが動作し続け通常業務に影響を与えることはありません。(ディスク・デュープレックス機能)
また、書き込みのたびに、そのセクターを読み出し、セクターに不良があるとわかったら、以後そこを使わないようにする不良セクター対策(Hot Fix機能)やOSにとって重要なディレクトリー・テーブルとファイル・アロケーション・テーブルについては、それぞれ別の場所に書き込んでおき、読み出しの際には、コピーと比較してエラー検査をします。(ディレクトリー構造の二重化、ファイル・アロケーション・テーブルの二重化など…)
以上のようにセキュリティと障害対策が他社製品に比べて非常に強力であることがNetWareの特徴といえます。

●NetWareとTCP/IPの共存が可能

TCP/IPプロトコル、PC/NFS機能によりUNIXマシンおよびSUNなどのワークステーションをサーバーにするシステムをサポート。これによってNetWareとTCP/IPソフトが同時に共存可能になりました。

●NetWare製品ラインアップ

「NetWare V. 2. 2英語版」 「NetWare V. 2. 2日本語版」
「NetWare 386 V. 3. 1J」 「NetWare 386 V. 3. 11英語版」
以上のNetWareシリーズを取扱っております。なお「NetWare 386 V3. 1J」に関しましては、接続可能なPCが他のNetWareと異なりますのであらかじめご確認ください。

世界の舞台で磨かれた
Acerファイル・サーバー Acer1100/33 (PC/AT互換機)をベース

- i80386(33MHz)CPU,RAM 4MB
- 5.25インチ(1.2MB)FDD×1基、
- 100MB~680MB HDD最大2基 実装可能
- 14インチモノクロモニター、モニターI/Fカード
- 16ビットバス用 Ethrnetカード1枚 標準装備
- フロアスタンド他

＊Net WareはNOVELL Inc.の登録商標です。

丸文株式会社

東京都中央区日本橋大伝馬町8-1 丸文ダイヤビル 〒10 第2営業本部 営業第3部 PC&ネットワークグループ TEL (03)3639-9897(ダイヤルイン)

待望のBORLAND C++ & APPLICATION FRAMEWORKS 3.0登場前に、
面倒無用のバージョンアップ先行予約。パッケージの「バージョンアップ・シール」に要注意!!

DOS上で、そしてWindows上で利用するアプリケーションの開発ツールとして、機能・スピード・価格面ともに理想的といえるC&C++コンパイラ。それが「BORLAND C++」です。今、このBORLAND C++のバージョン2.0をお買い上げになり、添付の無料バージョンアップ・シールを登録ハガキに貼ってご返送いただくと、ニュー・バージョン3.0の出荷時(4月下旬)に自動的に、しかも無料でバージョンアップのサービスをお受けいただけます。どうぞこのまたとないチャンスをお見逃しなく、ぜひご利用ください。

BC++ & APPLICATION FRAMEWORKS 3.0の主な追加・強化機能

■**コンパイルスピードがさらにアップ**

超高速なコンパイルスピードによって、開発時間をより短縮でき、開発者の生産効率を大幅に向上させることができます。一度コンパイルしたヘッダを使用し、無駄な時間を省くプリコンパイラヘッダ機能とあわせ、たいへん快適な開発環境を提供します。

■**Windows上で動作する新しい統合開発環境**

ソースコードの編集、コンパイル、実行といった開発作業がWindows上で行えるようになりました。この新しい開発環境には、マルチウィンドウのエディタ、オブジェクトの関係をグラフィカルに見ることのできるオブジェクトブラウザ、環境の機能に簡単にアクセスできるスピードバー、WindowsAPIのリファレンスを参照できるオンラインヘルプなど、実用的で扱い易い操作性がプログラマーをサポートします。

■**強力なオプティマイザ(最適化機能)を搭載**

ボーランドの新しいグローバルオプティマイザにより、アプリケーションを小さく、速くすることが可能になります。広範囲にわたる革新的な最適化の機能がコードサイズを最小にし、実行速度を最大にします。●Global register allocation ●Local and global common sub-expressions ●Induction variable analysis and strength reduction ●Loop invariant code motion ●Register parameter passing ●Dead code elimination ●Copy propagation ●Pointer aliasing ●Loop compaction ●Structure copy inlining ●Code compaction ●Redundant laod suppression ●Intrinsic function inlining

★BORLAND C++ 3.0では、286以上のCPUと拡張メモリ、ハードディスクが必要となります。適応機種はPC-9801シリーズ、PC-286/386シリーズです。

■**価格**……………… **¥98,000**(税別)

2.0一本分の価格で、プラス3.0もあなたのものに!

BORLAND® C++

登録ユーザーの方には、4月中旬にバージョンアップのご案内をお送りいたします。

※All Borland Products are trademarks or registered trademarks of Borland International, Inc. ©Borland International, Inc.
※Microsoft, Windowsは米国マイクロソフト社の登録商標です。その他、商品名は一般に各社の商標です。

ボーランド株式会社
〒151 東京都渋谷区笹塚1-64-8 笹塚サウスビル TEL.03-5350-9370(代)
※移転いたしました。

T1017777040985
〈資料請求番号P04〉
雑誌コード17777-04